Energy-aware fault-mitigation architecture for underwater vehicles

Noname manuscript No.
(will be inserted by the editor)

Energy-aware fault-mitigation architecture for underwater vehicles

Valerio De Carolis · Francesco Maurelli · Keith Brown · David Lane

Received: date / Accepted: date

Abstract Energy awareness and fault tolerance are important aspects for extending the autonomy levels of today's autonomous vehicles. With the aim of preparing the way for persistent autonomous operations of underwater vehicles this work focusses its efforts on investigating the effects of actuator failures, on an Autonomous Underwater Vehicle (AUV) capable of long-term inspection missions. This paper introduces an energy-aware architecture that by observing the use of the on-board resources is capable of detecting faults and monitors the performance of the thruster subsystem in modern AUVs. The effect is an increased autonomy level in presence of unexpected events like performance degradations or sudden failures. Moreover an important contribution of this work is to process the great volume of information, collected at the lower sensor levels, into operational parameters that can be treated by higher level modules. These parameters form part of an abstract representation of concepts and capabilities that are available at a given time during the mission's execution. Once this representation has been updated it is made available to the planning and execution components that can adapt the mission's behaviour using the most recent knowledge about the vehicle's state. To validate the proposed approach we evaluate our system on a real platform, Nessie VIII AUV, in both in real sea conditions and in a controlled test tank.

Keywords energy awareness · automatic fault-mitigation · fault tolerance · knowledge representation · autonomous underwater vehicle

V. De Carolis · F. Maurelli · K. Brown · D. Lane
Ocean System Laboratory, Heriot-Watt University
EH14 4AS, Edinburgh, UK
E-mail: valerio.decarolis@ieee.org

1 Introduction

Autonomous Underwater Vehicles (AUVs) are increasingly used for many tasks, including autonomous surveys, inspection, and on-board data processing in many application fields ranging from natural sciences to off-shore industry applications. Moreover the interest in intelligent platforms able to operate persistently and autonomously in dynamic environments and that are able to overcome possible problems is growing. As part of the European project FP7 PANDORA (Lane et al, 2012) and as a key focus for our research group, we set our efforts towards increasing the autonomy levels in AUVs. Our aim is to prepare the field for the concept of persistent autonomy. This means introducing new capabilities in the vehicle's architecture, allowing it to operate robustly (Maurelli et al, 2013), to detect faults and to adapt its plan (Cashmore et al, 2014) in order to cope with unexpected events and changes in the environment. On the other hand whilst aiming for extended runtime, it is increasingly important for these platforms to optimise the use of their on-board resources, for instance the energy stored inside the vehicle's battery or the amount of time needed for completing a complex mission.

This work introduces an energy-aware fault mitigation architecture to be used for augmenting the fault-tolerant capabilities of current AUVs. This is especially useful in presence of partial failures, like the performance degradation of actuators, where vehicle reconfiguration and task adaptation are valid alternatives to the interruption of the mission. This architecture differs from past proposals (Dearden and Ernits, 2013) by using the *energy usage* as the primary metric to assess changes in the vehicle's status. This is done at runtime using only on-board resources and avoiding interactions with human operators or remote systems. The proposed framework relies on a small set of parameters. This enables the use of a short procedure to tune the pro-

posed system after deploying the vehicle directly in the field. This represents another difference with other fault detection systems that often require extensive off-line training and a detailed knowledge of the platform to be tuned efficiently. Such behaviour makes complex architectures (Hamilton et al, 2001; Miguelaez et al, 2011) less suitable to be used in cases where changes in payloads and configuration are expected from different missions. This is often the case of low-cost inspection AUVs where payload sections are often adapted to mission requirements just before starting sea operations. Moreover, the use of a generic representation for the health status and failures of actuators allows this framework to be implemented on a broad set of vehicle designs. Furthermore, this can be extended with a small effort to more complex designs where control fins or rotatable thrusters are often employed.

Together with fault mitigation this work focuses on the concept of energy-awareness and shows its relationship with the overall vehicle's architecture. This highlights the higher level implications of actuator failures in the planning and reasoning sub-systems often found in modern designs. This represents another difference with past architectures where the use of on-board resources is often not taken into account. Aiming at long-term operations this system allows tracking of the vehicle's behaviour over two different time frames: a short one, related to the current execution task (i.e. the inspection of an object or a structure) and a long one, related to the entire length of the autonomous mission. An automatic fault mitigation is introduced over the short time frame allowing the vehicle to complete its current task. The effects of detected failures are then evaluated at higher levels over the longer time frame by using the information collected in a knowledge graph.

This paper is organised as follows: Section 2 introduces the related work done in the area of component modelling, diagnostics and fault-tolerant control as well as the role of fault management, Section 3 presents the approach used in this work for detecting the failure of thrusters, the fault mitigation solution used to cope with the detected faults and the representation of knowledge about the internal status of the vehicle, Section 4 presents the experimental results collected using the research vehicle Nessie VIII AUV in different operational scenarios and Sections 5 and 6 discuss the contributions described in the paper.

2 Related Work

In the field of underwater vehicles, and in robotics in general, long-term autonomy and robust autonomous operations are active areas of research. One example is the concept of *persistent autonomy* where an autonomous underwater platform is capable of executing long-term missions while coping with some uncertainty instead of relying on the complete knowledge of the operational environment (Lane et al, 2012). With this goal in mind, several aspects may be taken into account when improving current architectures: adaptive planning, knowledge representation, status assessment and platform management are all elements that can improve the effectiveness of autonomous vehicles in unknown scenarios. There is much current work ongoing in these areas and each of them carries a relevant contribution for improving the autonomy levels in the domain of underwater vehicles (Insaurralde and Lane, 2012) (Seto, 2012).

In the context of diagnostics and fault detection for AUVs, for instance, relevant work has been done focusing on the thruster subsystem. Several methods have been proposed for deriving an accurate thruster model by means of analytical modelling and experiments in controlled environments. The work proposed by (Healey et al, 1995) relies on the analytical modelling of the principal effects that contribute to the real behaviour of marine thrusters, suggesting a three element model (electric motor, propeller mapping and fluid model) and a set of identification experiments in order to derive the final model. This, while it is extensive, requires a great number of parameters to identify and relearn if a modification is done in the thruster configuration (for instance in terms of nozzles or tunnels). The work of (Kim et al, 2005) develops the early works considering the effect of the ambient flow velocity and suggesting a performance drop in non-static conditions with respect to previous analysis. In this case, while improving the modelling with respect to early works, the need of a controlled environment where the flow and vehicle velocity can be assumed to be constants makes it impractical for an on-board implementation inside a modern AUV.

These works, while offering an effective procedure to derive and validate the parameters used in such models, focus on the detailed analysis of thruster behaviour under different environmental conditions without taking into account the effect of control electronics often embedded in such components. These are, for instance, electronic speed control (ESC) circuits employed in many brushless DC marine actuators. An ESC act as a proxy between the electrical motor itself and a higher control architecture adjusting the component's behaviour according feedback sensors. In that case vehicle's designer has little capability of modelling its accurate behaviour and a black-box approach is often more appropriate to overcome those limitations.

Aiming at a simpler modelling strategy, the work (Hanai et al, 2008) suggests an effective procedure that can be followed is to derive a model which relies on fewer parameters. This is thus better suited to be implemented inside the control software without requiring a great use of computational resources. This work, however, does not explicitly take into account the effects of delays and latencies but yet propose a suitable approach for identifying the actua-

tor's characteristic. Additional aspects such as delays and latencies, while not of primary importance from the modelling point of view, gains their relevance if the resulting model is meant to be used inside a real-time diagnostic system embedded in a greater software architecture operating in a resource constrained environment. In developing this work we decided to follow a data-driven approach for deriving models of actuators used in our research platform. This allows the extraction of a mathematical model directly from the data collected by platform itself before starting a long-term mission. This approach is useful when in presence of commercial components, where some of the internal aspects are not disclosed by the manufacturer, and a black-box approach need to be used. Once models have been derived they are re-used for other functions inside the proposed control scheme, for instance as a linearisation term of the thrust to throttle relationships.

Several strategies can be implemented to build a diagnostic system that monitors the vehicle's internal status and its actuators. Previous work has been done in this area, showing that the runtime detection of faults is possible and similar strategies have been applied to remotely operated vehicles (Corradini et al, 2011) and marine vessels (Cristofaro and Johansen, 2014). It has also been shown that in presence of redundancy in the thruster configuration even an adjustment can be introduced allowing the vehicle to resume its operations. Most of these strategies try to derive the health status of the on-board actuators by looking at common control and feedback signals, like input voltages and commands, propeller speeds and actuators' currents. They differ, however, in the way they extract the information about the presence of faults, by the number of parameters required to tune the system and by the presence of an extensive validation over real missions. The work of (Antonelli et al, 2004) is based on the use of support vector machines (SVM) as the core functionality for the diagnostic system and it is able of exploit the robustness of a machine learning approach in dealing with some degree of uncertainty. This, in fact, while being able to learn from previous experience, requires the use of a clean and an extensive dataset together with an off-line training procedure. The contribution of (Mišković and Barišić, 2005) is based, instead, on principal component analysis (PCA). This offers the interesting feature of dealing with a broad range of faults while recognizing their typology based on some logical relationships. This strategy, like the one discussed above, has been presented together simulation results. It implementation relies on the availability of a precise statistical model for the control input signals. These aspects along with the lack of extensive experimental validation makes this method less suitable for implementation. The work of (Wang, 2012) suggests the use of fuzzy neural network to model the vehicle's behaviour and to detect the presence of a fault if its navigation deviates the expected behaviour. This is accomplished by applying a residual detection between measured data and model output. This approach, while flexible because of its fuzzy representation, presents only simulated results and, like other machine learning approaches, relies on the availability of good training data, in this case not for a specific component, but for the motion behaviour of the complete vehicle. While of interest its application is out of the scope of this brief.

The work of (Alessandri et al, 1999; Hanai et al, 2009), on the other hand, present the use of a simpler yet more effective model-based diagnostic strategy. In the first case vehicle dynamics are taken into account and a diagnostic loop is closed around the vehicle's motion. A series of filters then evaluate its behaviour under different fault hypotheses and a residual evaluation is then employed to detect the presence of a specific fault condition. In the second case general dynamics are not taken into account and a detailed analysis of a model-based thruster fault detection scheme is presented. The actuator's model is derived through a series of experiments and real measurements. In this case only the relationship between input commands and feedback signals is employed. This provides an effective way to detect deviations from the actuator's nominal characteristic, something more difficult to detect if taking into account a larger motion model for the complete vehicle. These works, while of relevance for this brief, do not discuss the integration of proposed methods with the greater vehicle's software architecture and how the information derived from fault detection can be re-used to overcome the presence of failures.

Other fault detection strategies (Isermann, 2005a) rely on model-free methods, such as fault-tree analysis, spectrum analysis, limit checking or, more in general, qualitative approaches. While still of relevance from a diagnostic point of view their application to the AUV domain requires extensive tuning, expert knowledge and parameters that limits their scope to a specific vehicle's configuration. This is in contrast to on-the-fly adjustments and automatic tuning capabilities often researched in the context of long-term autonomy operations. For these reasons their analysis is outside the scope of this paper.

Despite the availability of different strategies the experimental validation (Hanai et al, 2009) and operational experience acquired during Arctic missions (Caccia et al, 2001) suggests that a model-based approach represents the best candidate for building a real-time diagnostic system without introducing much computational complexity in the software architecture of a modern AUV. This approach offers an accuracy proportional to the quality of the underlying component model used to derive the diagnostic metric. Also it does not require a great number of parameters that need tuning according to the specific type of underwater missions. This gives model-based strategies the required robustness and solidity to be implemented in a long-term autonomy

architecture. With these aspects in mind this work implements a model-based diagnostic strategy to be used together with an existing energy measurement framework (De Carolis et al, 2014) developed during previous works. This choice allows the proposed subsystem to rely on a minimum set of input signals while taking advantage of energy usage measurements calculated at runtime on-board of the vehicle.

Beside the fault detection and isolation (FDI) aspects, other relevant work has been done in the area of fault accommodation. This takes care of changing the vehicle's configuration to mitigate the effect of detected failures. The feasibility of such approach is highly affected by the vehicle's design, for instance, in the case of thrusters the presence of redundancy. An example is the X-shaped thruster configuration, where a 45 degrees offset from the vehicle's principal axis offers a good flexibility even in presence of thruster failures. In such cases an alternative *thrust allocation solution* may be found by remapping the distribution of forces among the available thrusters. This allows the restoration of (at least to a given extent) the control of degrees of freedom (DOFs) present in the original scheme. This approach has been proposed in (Yang et al, 1998) where an experimental study has been conducted relying on the redundancy of the vehicle's thrusters configuration. Later, the work of (Sarkar et al, 2002; Omerdic and Roberts, 2004) reviewed this strategy with the introduction of weighting coefficients in a more generic formulation. This enables it to proportionally redistribute forces among all the available actuators according to their weights. The concept of *thruster efficiency* is then introduced and coupled with a diagnostic framework in the contribution of (Hanai et al, 2010). In this work the diagnostic system is providing a health status estimate of the vehicle's actuators while dynamically adjusting the redistribution weights among thrusters in presence of faults. In the case of work developed on ODIN AUV (Sarkar et al, 2002) the introduction of weighting requires the use of a modified controller schema to prevent thruster saturation. This aspect, while solving elegantly the problem of saturation, introduces some restrictions in the design of customized control architectures. An example of this is seen in (Karras et al, 2013, 2014) where a common control schema has been adapted to a specific operating mode that an AUV may encounter during its long-term operations. In such cases the fault adaptation should be attempted even if in presence of control strategies, selected according to the current task. Furthermore, in the work developed on the SAUVIM AUV (Hanai et al, 2010) the integration of a fault accommodation system with the vehicle's software architecture is not discussed. Two other aspects are also left aside in the above works: the effects of force redistribution on the navigation performance and on the energy usage profile (Willcox et al, 2001) of the autonomous platform. These aspects, while not of primary importance from the control point of view, influence the behaviour and effectiveness of vehicles over the duration of long mission. Incorporating these concepts into a unified architecture can help decisions taken by higher level autonomy modules for instance at the planning and the mission execution levels. This, which represent an active area of research (Cashmore et al, 2014), enables modern autonomy architectures to adapt and optimise on the fly the current execution plan in terms of resource usage and task selection during long-term missions.

Other fault tolerant techniques exist and do not rely on classical control methods. In those cases mitigation solutions are found using machine learning or evolutionary methods (Wang et al, 2006), often relying on the availability of training data. An example of *reinforcement learning* approach is shown in (Ahmadzadeh et al, 2014a,b) where a new control policy is learnt in case of actuator's loss. Such strategy, while effective in recovery from hard faults, requires a great deal of simulation and experimentation before a valid policy is derived for a specific fault condition. Such a constraint is not always applicable for vehicles operating out at sea where immediate reactions are needed, for instance to avoid dangerous behaviours, or when a minor failure or degradation is detected.

Moreover, work in the area of integrated health management and control has been done in the context of ground (Tang et al, 2011) and aerial (Ge et al, 2004) autonomous robots. In these domains availability of cheap sensors and high communication bandwidths allow designers to develop architectures where some of the diagnostics and adaptation modules are not implemented inside the vehicle itself. In the underwater domain, on the other hand, the limitations of acoustic communications require the development of such architectures inside the vehicle itself (Hamilton et al, 2001) where most of the sensors data has to be processed on-board in order to extract decisions that can be optionally shared over the low-bandwidth communication link (Miguelaez et al, 2011) to inform neighbours vehicles and human operators.

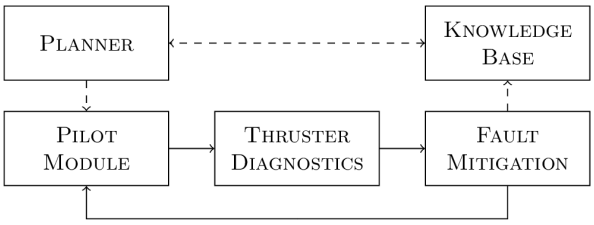


Fig. 1: Overview of data-flow in the fault mitigation system. This schema shows the two feedback loops that acts on the control software after detecting the presence of thruster faults. The knowledge loop is considered asynchronous and thus acting at a slower rate than the diagnostic one.

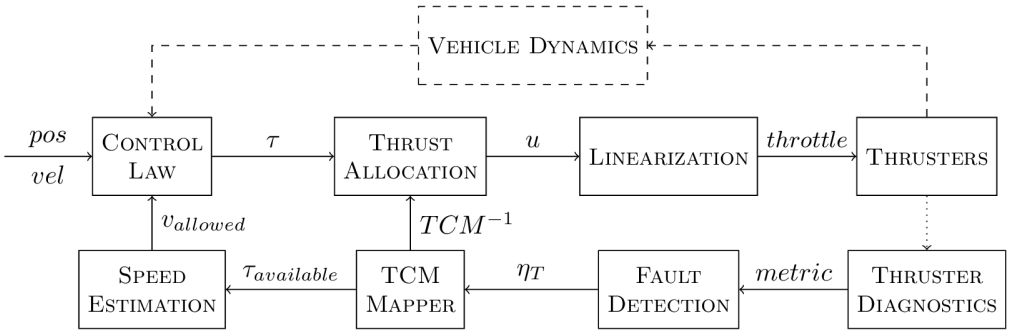


Fig. 2: Schema of the fault-tolerant control architecture for Nessie VIII AUV. The fault mitigation algorithm estimates at runtime the thruster efficiencies η_T and the allowed cruise speed for the actual configuration $v_{allowed}$. This influences the control law which takes into account any limits on the navigation speed when computing the control forces τ to move the vehicle. Moreover the fault-mitigation algorithm calculates an updated version of the *thruster configuration matrix* (*TCM*) used for allocating the control forces on the available actuators.

3 Experimental Framework

As outlined in the previous sections an automated fault mitigation architecture relies on different components that when combined together are used to model, detect and react to the presence of failures in the vehicle's actuators. The introduction of these components allows the addressing of one of the key aspects of long-term autonomy operations: the survival to unexpected situations without interaction with human operators. In this work this is achieved by adding few software modules to an existing vehicle's architecture. These monitor the internal state of the platform and adapt its behaviour during an autonomous mission. Upon the detection of an unexpected event the proposed framework introduces an immediate reaction, adjusting the platform's configuration and allowing the current task to be completed. Concurrently a longer term response is researched through the use of a high level knowledge representation. This relates individual concepts like components, capabilities and elementary actions to propagate the consequences of detected failures from the current task to the overall mission.

A overview of the proposed system is shown in Fig. 1. The central component is a *model-based diagnostic* module. This is monitoring the input commands and the measured *energy usage* of each actuator. The choice of an energy-based metric rather than multiple diagnostic signals allowsthe use of a smaller set of values while still detecting the presence of faults. The *energy usage* is derived at runtime using measurements provided by existing sensors often found in modern designs, such as voltage meters and current probes fitted inside the vehicle or its actuators. Together with these other reliable metering systems are found in existing battery operated platforms, such as battery management systems (BMSs) used to monitor the health status of on-board battery packs and their residual charge. In these cases such sensors can be re-purposed to be used with the energy-driven diagnostic module. Another component is the *fault mitigation* algorithm. This reconfigures the control software (*pilot module*) upon the detection of a fault. This is achieved with the use of a modified control schema, shown with more details in Fig. 2, that allows to adjust the use of available actuators at runtime. After a mitigation has been introduced the updated status of the thruster system is propagated inside a *knowledge base* (KB), another key componet of the proposed architecture. The KB allows high-level autonomy modules, like a *planner*, to adjust the remaining part of a mission if a better strategy can be computed. This is influenced by the severity of detected faults and, in extreme cases, could require the use of different navigation modes, in terms of trajectory planning or motion control, if failures can disrupt normal navigation capabilities.

In the following sections individual components are discussed, highlighting their role and contribution to the assessment of operative conditions in presence of faults.

3.1 Thruster Model

The first module is the *Thrusters* component, shown in Fig. 2. This takes care of modelling the operations of vehicle's actuators. It takes input commands from the controller subsystem and produces output currents that represent their nominal power consumption. As outlined in (Hanai et al, 2009), defining exactly the behaviour of marine thrusters is a difficult task, mostly because of their non-linear characteristics and deviations from an ideal behaviour over their lifetime. Although other techniques exist (Healey et al, 1995; Kim et al, 2005), we decided to use a data-driven approach for deriving the thruster model used in this work, shown in Fig.

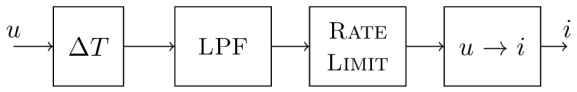


Fig. 3: Thruster model schema. The *throttle to current* characteristic $u \rightarrow i$ is applied after introducing the communication delay (ΔT) and non-linear effects of internal speed controllers of a generic brushless DC motor.

3. This is because models that take into account the coupling effects among electric motor, transmission shaft and the actual propeller often rely on multiple parameters that, in the presence of commercial grade thrusters, are often undisclosed by the vendors or difficult to estimate without knowledge of their internal characteristics. With these constraints in mind we integrated a training procedure inside the modified software architecture. This requires the vehicle to stimulate its actuators using known input signals while recording their behaviour. This collects a calibration dataset used for calculating the *thruster model* and extracts its nominal characteristic. The thruster input command u or *throttle* is varyied across the allowed range a with a repetition period p. Each input command is held steady enough to allow each thruster to reach and settle to the requested throttle configuration . The signals used in this procedure are: a step function (1), a triangular signal (2) sweeping between full-forward and full-reverse commands and a slowly changing sinusoidal one (3) with full range amplitude and period comparable to the update rate of the control architecture used in the vehicle.

$$u_{step}(t) = a\,\text{step}(t-T) \tag{1}$$

$$u_{trig}(t) = \frac{2a}{\pi}\arcsin\left(\sin\left(\frac{2\pi}{p}t\right)\right) \tag{2}$$

$$u_{sin}(t) = a\sin\left(\frac{2\pi}{p}t\right) \tag{3}$$

After recording the training data a regression procedure is employed to extract the *throttle to current* actuator's characteristic used in the rest of this framework. Together with that parameters, such communication delay and device latency, are estimated from a time analysis of signals (1) and (3).

This procedure is known as Locally Weighted Project Regression (LWRP) (Vijayakumar et al, 2005), a state-of-the-art non-linear regression algorithm. This employs multiple linear models to approximate, on a smaller domain, high dimensional non-linear functions. The use of this approach allows the method to represent efficiently the non-linear behaviours often found in the underwater domain such as motion models (Fagogenis et al, 2014) or, in this case, the characteristics of marine actuators. The LWPR allows for incremental on-line learning, through the introduction of a forgetting factor, and it is best featured when a good number of samples (≥ 2000) are available. This property makes LWPR a good candidate for the use of case introduced in this work. For instance, this allows the adjustment the extracted models several times over the lifetime of an AUV to compensate for ageing effects of actuators by collecting fresher data using the training procedure previously described. The LWPR algorithm defines the sub-domains in which local models are calculated as receptive fields (RFs). These are defined using Gaussian kernels. In RF the output function is approximated using a lower dimensional linear model calculated using Partial Least Squares (PLS) fitting. The final result of the target function $\hat{y}$ is computed using the weighted sum of the predictions of each local model:

$$\hat{y} = \frac{\sum_{k=1}^{K} w_k \hat{y}_k}{\sum_{k=1}^{K} w_k} \tag{4}$$

$$w_k = \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{c}_k)^T \mathbf{D}_k (\mathbf{x}-\mathbf{c}_k)\right) \tag{5}$$

where the weights w_k are taken from the kernel of each receptive field, $\mathbf{c}_k$ represents the centre of k-th receptive field, $\mathbf{D}_k$ the inverse covariance of receptive fields, K their total number, $\hat{y}_k$ their local prediction and $\mathbf{x}$ the input query point. A detailed description of the algorithm is outside the scope of this paper and further details can be found in (Vijayakumar et al, 2005). The algorithm's parameters are researched during its *training* phase and related to the input dataset through the use of a Mean Squared Error (MSE) metric to evaluate the model's predictions.

In this framework the extracted characteristic is represented analytically as:

$$i(t) = f_m(u_{lim}(t)) \tag{6}$$

where i is the current used by the thruster, u_{lim} the input throttle and $f_m(.)$ is the extracted LWPR model. The u_{lim} term is derived from the raw control input u taking into account the effect of communication delay ΔT:

$$\Delta u = u_d(t) - u_d(t-1) \qquad u_d(t) = u(t-\Delta T) \tag{7}$$

where Δu represent the difference between consecutive commands and the non-linear behaviour of ESC circuits through the use of a rate limiter:

$$u_{lim}(t) = \begin{cases} u_d(t) & \text{if} \quad \Lambda_{low} \leq \Delta u \leq \Lambda_{high} \\ u_d(t) + \Lambda_{high} & \text{if} \quad \Delta u > \Lambda_{high} \\ u_d(t) - \Lambda_{low} & \text{if} \quad \Delta u < \Lambda_{low} \end{cases} \tag{8}$$

where Λ_{low} and Λ_{high} represent the maximum rate of change of throttle command u. These parameters are numerically estimated using the *average rate of change* (RC) for the training signal (1):

$$RC = \frac{u_d(t+T_h) - u_d(t)}{T_h} \tag{9}$$

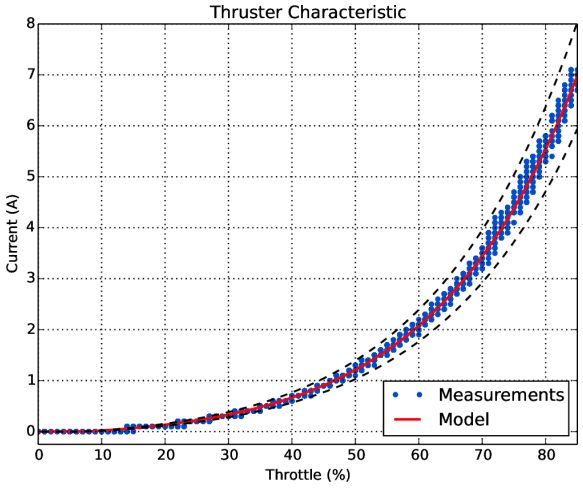


Fig. 4: Thruster characteristic for the port-side forward actuator of Nessie VIII AUV. Its behaviour is highly non-linear with most of the power being developed at high throttle requests.

by taking into account the time T_h needed to correctly changing the throttle setting between 5% and 95% of the allowed ranges $(0,a]$ and $[-a,0)$.

An example of real characteristics is shown in Fig. 4 for a Seabotix HPDC1502 thruster. It is calculated operating in a controlled environment using the feedback current signal from its embedded controller. This has a low noise output but a reduced precision of 0.1A yet it allows to extract a good quality model for most of the thruster's operative range. Additional precision can be achieved using external metering sensors.

3.2 Thruster Diagnostics

The second module of the proposed architecture is the *Diagnostics* component. It combines data generated by actuator's models with measurements collected at runtime. This calculates a diagnostic metric representing deviations from ideal behaviours that will be used to detect the presence of failures in a *fault detection* module.

In developing this work a *model-based* diagnostic approach has been implemented. The schema described in this section is shown in Fig. 5. This real-time module produces a zero output metric if actuators are operating following their nominal characteristics. The metric m is derived from a feature called *short-term energy* (STE). The feature $\varepsilon \geq 0$ is calculated by integrating the energy usage of a given actuator over a short time window W_e. This choice has been suggested by the results of early experiments with an integrated energy measurement framework (De Carolis et al, 2014). The STE can be written in discrete time for both the

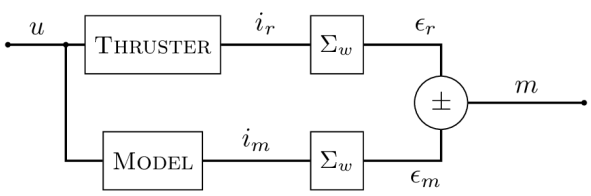


Fig. 5: Schema of the diagnostic subsystem. The u signal represent the actuator commands sent by the *pilot module*. The model estimates the current usage i_m for each input while dedicated sensors measure the usage i_r of the real component. Two integrators then calculates the STE features $\varepsilon_m, \varepsilon_r$ and a comparator calculates the diagnostic metric m used for detecting the presence of faults.

real actuator and its model as:

$$\varepsilon_j[n] = \sum_{k=0}^{W_e} v_{nom}\, T_s\, i_j[k] \qquad j \in \{r,m\} \tag{10}$$

where v_{nom} is the thruster nominal voltage, T_s the sampling time, i_r the measured current and i_m the one calculated using the thruster model derived in the previous section.

The diagnostic metric m is calculated as sum of residuals $\Delta\varepsilon$ of short-term energy features normalized and filtered using an exponential smoothing over a time window $W_f > W_e$.

$$\Delta\varepsilon[n] = \varepsilon_r[n] - \varepsilon_m[n] \tag{11}$$

In this expression ε_r represent the measured short-time energy and ε_m the same feature derived from the thruster model's output. The metric m is thus given by:

$$m = \sum_{l=0}^{W_f} w_l \left(\frac{\Delta\varepsilon[l]}{\Delta\varepsilon_{max}} \right) \qquad m \in [-1,1] \tag{12}$$

$$w_l = \frac{e^{-\alpha l}}{\sum_j^{W_f} e^{-\alpha j}} \qquad \sum_{l=0}^{W_f} w_l = 1 \tag{13}$$

where w_l are exponential weights given the parameter α, which controls the smoothing, and $\Delta\varepsilon_{max}$ is the maximum value of the short-term energy residual. This is derived taking into account the characteristics of underlying actuators assuming one terms of (11) as maximum and the other as zero. Analytically:

$$\Delta\varepsilon_{max} = W_e\, v_{nom}\, T_s\, i_{max} \tag{14}$$

where i_{max} represent the maximum designed current usage of the device and W_e is the length of the integration window. Quantities W_e and W_f are chosen as the trade-off between accuracy and latency. They are considered as parameters for tuning the responsiveness of the proposed diagnostics module. In evaluating this work these have been set using experimental knowledge from early trials as following: $W_e = 2$ s, $W_f = 20$ s and $f_s = 1/T_s = 10$ Hz as sampling frequency.

3.3 Fault Detection

The third module is the *fault detection* component. This analyses the diagnostic metric provided by the *diagnostics* module and detects the presence of faults. The output is a sequence of decisions d that signal the detection of an underlying problem with a specific actuator. Moreover, this module keeps track of efficiency estimations for the thruster subsystem, allowing other modules (e.g. *mitigation*, *knowledge base*) to reason and adapt in presence of specific fault conditions.

One common method to detect the presence of a fault given the computed metric is the use of a threshold. This uses detection theory and derives the threshold's value to generate binary decisions with given performances in terms of probability of false alarm P_{fa} (e.g. assuming the presence of faulty condition in a fault free case).

$$d = \begin{cases} 0 & \text{if} \quad |m| < \lambda_d \\ 1 & \text{if} \quad |m| \geq \lambda_d \end{cases} \tag{15}$$

Different strategies can be used for deriving the λ_d threshold. One approach is to assume the metric is a random variable and express the generic λ_d threshold as:

$$\lambda_d = \mu_m + k\sigma_m = k\sigma_m \qquad k = 1,\ 2,\ 3,\ \ldots \tag{16}$$

where k is a scaling parameter, μ_m and σ_m the metric's mean and standard deviation. In a fault free case the mean is assumed to be zero while the term σ_m takes into account the residual uncertainty from real-time measurements noise σ_{sens} and the precision of the extracted LWPR model σ_{model}.

$$\sigma_m^2 \propto \sigma_{sens}^2 + \sigma_{model}^2 \tag{17}$$

If the metric m is assumed to be following a normal distribution $N(\mu_m, \sigma_m^2)$ the probability of false alarm is given by:

$$P_{fa} = \int_{\lambda_d}^{\infty} \frac{1}{\sqrt{2\pi\sigma_m^2}} e^{-\frac{x}{2\sigma_m^2}} dx \tag{18}$$

This suggests a straightforward procedure to calculate the value of threshold λ_d as function of the chosen P_{fa} and the uncertainty of the diagnostic metric σ_m. While developing this work the performance of detection system has been tuned to allow a probability of fault alarm of 1% or $P_{fa} = 0.01$. This choice while conservative, allows the system to reject outliers (e.g. given by synchronization errors) without reducing the system's responsiveness in presence of the degradation faults shown in the experimental section.

Under these assumptions the residual term $\Delta\varepsilon$ can be seen as difference between two normal distributions, respectively, $N(\mu_{sens}, \sigma_{sens}^2)$ and $N(\mu_{model}, \sigma_{model}^2)$ generated from the measurement and model process of a given actuator. In an ideal case, neglecting the model bias and assuming a fault free scenario, their means can be assumed to equal $\mu_{sens} = \mu_{model}$. This lets the $\Delta\varepsilon$ term be represented with a normal distribution of zero mean and known variance: $N(0, \sigma_{sens}^2 + \sigma_{model}^2)$. Taking into account these aspects the (12) can be rewritten as:

$$m = \sum_{l=0}^{W_f} w_l X_l \tag{19}$$

where $X_l = \Delta\varepsilon[l]/\Delta\varepsilon_{max}$. Under the assumption of independence between residual samples the variance of the metric can be expressed as:

$$\begin{aligned} \sigma_m^2 &= \mathrm{Var}\left(\sum_{l=0}^{W_f} w_l X_l\right) = \sum_{l=0}^{W_f} \mathrm{Var}(w_l X_l) = \sum_{l=0}^{W_f} w_l^2 \mathrm{Var}(X_l) \\ &= \sigma_x^2 \sum_{l=0}^{W_f} w_l^2 = \frac{\sigma_{sens}^2 + \sigma_{model}^2}{\Delta\varepsilon_{max}} K_w \end{aligned} \tag{20}$$

where K_w represents the scaling term given by the w_l coefficients. The terms σ_{sens} and σ_{model} are numerically estimated from (10) using the training dataset employed to extract the actuator's model:

$$\sigma_{sens}^2 \propto W_e\ v_{nom}\ T_s\ \sigma_i^2 \tag{21}$$

$$\sigma_{model}^2 \propto W_e\ v_{nom}\ T_s\ \sigma_{lwpr}^2 \tag{22}$$

where σ_i^2 represents the precision of current measurements (e.g. $\pm 0.05\ A$ for the validation platform) and σ_{lwpr}^2 is given by the confidence intervals of the calculated LWPR model.

The choice of the proposed approach is motivated by the operational experience on an experimental platform as well as the availability of a single fault hypothesis which allows a probabilistic analysis of the metric. Furthermore, experimental results shown in the following sections present satisfactory levels of performance when testing the system on a real AUV. Beside the use of a single threshold other fault detection techniques have been successfully applied in the context of fault detection such as the use of Artificial Neural Networks (ANN) (Isermann, 2005b) or the use of Fuzzy Decision Making (FDM) methods (Mendonça et al, 2009). The FDM approach represents a relevant technique as it allows to detect and isolate multiple faults using the same type of residuals presented in this section. Fuzzy methods are useful in the presence of noisy or imprecise measurements and when interpretation is highly dependent on human experience. In the case of FDM the use of simplified membership functions, aggregation operators supports a detection system using a common threshold for multiple fault hypothesis. The use of FDM is considered for the future work related to the proposed energy-aware architecture.

3.3.1 Fault Model

In the marine environment thruster failures can be related to several causes (Antonelli, 2006), like the presence of objects

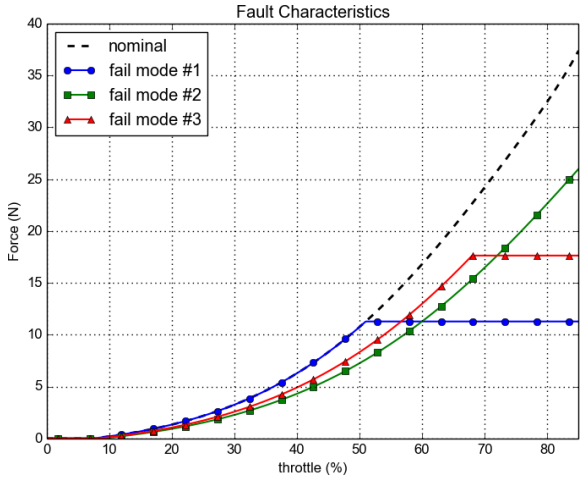


Fig. 6: Example of different fault characteristics. In this first case the thruster is operating with a limit on its maximum power, capping its effort above 50% of the input command. In the second case a reduced thrust efficiency ($\eta_T = 0.8$) is introduced. In the last case the combined effect of the previous modes is shown with $\eta_T = 0.9$ and capping above 68%.

blocking the propeller, rope tangling or water leaks that can disrupt the availability of the actuator itself. In these cases operational experience (Caccia et al, 2001) as well as other studies on this topic suggests that, beside the complete loss of the actuator, the *performance degradation* failure mode is a relevant problem that need to be addressed during real sea operations. This mode, as the name implies, is related to the loss of efficiency in the failed component, which produces a different effect on the vehicle than the one expected by its control architecture. This fault, often difficult to detect in presence of feedback control loops, can be modelled as an actuator that presents a *thrust efficiency* η_T lower than one.

$$\eta_T = \frac{T_{real}}{T_{nominal}} \qquad 0 \leq \eta_T \leq 1 \tag{23}$$

This is defined as the ratio between effective thrust T_{real}, experience during real operations, and the design thrust $T_{nominal}$, achievable in ideal conditions. The parameter η_T can thus be used to represent the health status of the actuator. In this framework the severity of a degradation is controlled by adjusting η_T: from the complete loss of efficiency ($\eta_T = 0$) to standard operative conditions ($\eta_T = 1$).

In developing this work we focused our scope on the effect of degradation $\eta_T > 0$ rather than the complete loss of the actuator. In these cases the vehicle is supposed to use the *degraded thruster* without removing it from the solution of thrust allocation problem. This is until the high-level autonomy modules suggest which longer-term adaptation needs to be implemented. A total failure $\eta_T = 0$, while still relevant, can lead to the loss of a DOF if no redundancy is available in the vehicle's design. This is the case for the *heave* DOF of the vehicle considered in the rest of this paper. Because of these aspects experiments along that DOF are not taken into account. If such loss is detected an alternative solution, which does not rely solely on the remapping of control forces, can be researched, for instance, by adapting the vehicle's motion and trajectory (Ahmadzadeh et al, 2014a,b) to compensate for the lack of control along a specific axis. Discussion of such approaches is outside the scope of this brief.

The resulting characteristic of a degradation fault is often unknown and only a qualitative description can be estimated. In Fig. 6 some of the possible characteristics are represented. In all of those cases the actuator's behaviour is disrupted and the nominal thruster characteristic is not followed along the full range of input commands. From a low-level point of view the degradation fault can be seen, for instance, as the effect of a damaged ball-bearing, propeller or nozzle which affect the actuator's performances. Analytically this characteristic can be represented as:

$$T_i(u) = \eta_{T_i}\, G_i(u) \qquad i \in \{0,\ 1,\ \ldots\ ,\ N\} \tag{24}$$

where G_i is the *throttle-to-thrust* relationship for the i-th thruster, η_{T_i} its efficiency and N the number of thrusters. The $G(u) \propto f_m(u)$ term can be derived from (6) with the use of the relationship between electrical power and output thrust[1].

In developing this work the first failure mode is targeted explicitly. Its effects are reproduced by limiting the maximum power output of a target actuator. In the vehicle this is achieved by manipulating on-the-fly the input commands to reduce its output thrust. This procedure is known as *fault injection*. Once such modification is in place the actuator's characteristic deviates from its standard condition triggering, then, detections in the diagnostic system. This behaviour has been observed also in presence of a real failure (e.g. the loss of a propeller blade) as will be detailed in the experimental section. Despite this focus, other failure modes could be handled by the proposed framework. This feature is given by the choice of an energy metric that highlights the presence of a faulty behaviour as long as a change in the energy usage is detected.

The severity of detected failures is represented using a qualitative approach. This allows other modules, such as the fault mitigation, to react to generic fault conditions and yet implement a possible correction for a short time horizon. For this reason a counting framework has been also introduced in the proposed architecture. This translates the series of diagnostic decisions for each actuator into an estimation of their thrust efficiency. This is assumed to be optimal at the beginning of a mission and, in presence of a sequence of positive detections for t_f consecutive time instants, it is

[1] For commercial thrusters, like the one used in this work, this is given by the thruster's manufacturer. Alternatively it can be calculated experimentally with any of the methods discussed in Section 2.

gradually decreased until the actuator's functionality is restored (e.g. the metric falls below the λ_d threshold) or its use is declared not feasible from an higher level reasoning module.

$$\eta_{T_i}[n] = \begin{cases} \eta_{T_i}[n-1] - \delta_{\eta_T} & \text{if fault detected} \\ \eta_{T_i}[n-1] & \text{if nominal} \end{cases} \tag{25}$$

The recovery capability is modelled by introducing a coefficient $\gamma_{\eta_T} \ll \delta_{\eta_T}$ that allows the system to restore the use of a given actuator if no further mismatch is found after a specific time $t_r \gg t_f$ during normal navigation and its current efficiency is above the 0.2 threshold. This choice allows the system to cope with transient faults, such as the ingestion of sea weed in tunnel thrusters or debris in marine actuators. The settling time t_r is usually specific to a particular type of thruster nonetheless, in case of severe failures (when $\eta_{T_i}[n]$ falls below 0.2), the faulty actuator is first excluded from the system (e.g. to prevent further damage) and a notification is escalated to the higher level knowledge base.

$$\eta_{T_i}[n] = \begin{cases} \eta_{T_i}[n-1] - \delta_{\eta_T} & \text{if fault detected} \\ \eta_{T_i}[n-1] & \text{if nominal} \\ \eta_{T_i}[n-1] + \gamma_{\eta_T} & \text{if fault cleared} \end{cases} \tag{26}$$

3.4 Fault Mitigation

The fourth module is the *fault mitigation*: this groups the smaller components, *thrust allocation*, *tcm mapper* and *speed estimation*, these are shown in Fig. 2. These process the output of *fault detection* module (e.g. estimated efficiencies and binary decisions) to introduce a short-term reaction to unexpected failures during the execution of autonomous tasks. Their output is a set of estimated quantities (e.g. allowed speed, available forces) that are shared with the higher level *knowledge base* module. At the same time these adjust the internal behaviour of the control subsystem, for instance selecting a different allocation procedure (i.e. in case of saturation), to maintain satisfactory navigation capabilities.

After detecting a fault the proposed system implements a mitigation by adjusting the use of available actuators according to their estimated efficiencies η_{T_i}. A well-known approach to achieve this is to modify the thrust allocation policy by updating the *thruster configuration matrix* (TCM) commonly used in the control schema of underwater vehicles. The TCM matrix, which depends on the positions $e_i = [e_{x_i}\ e_{y_i}\ e_{z_i}]$ and the orientations $r_i = [r_{x_i}\ r_{y_i}\ r_{z_i}]$ of each thruster with respect to vehicle's center of mass, $C = [x_c\ y_c\ z_c]$, it is defined as:

$$B_{(6\times N)} = \begin{bmatrix} e_{x_1} & \dots & e_{x_N} \\ e_{y_1} & \dots & e_{y_N} \\ e_{z_1} & \dots & e_{z_N} \\ (r_1 \times e_1)_x & \dots & (r_N \times e_N)_x \\ (r_1 \times e_1)_y & \dots & (r_N \times e_N)_y \\ (r_1 \times e_1)_z & \dots & (r_N \times e_N)_z \end{bmatrix} \tag{27}$$

It is used by the control software to distribute the requested force $\tau_{(1\times 6)} = [x\ y\ z\ k\ m\ n]$ with respect of the centre of mass to individual thrusters forces $f_{(1\times N)} = [f_0\ \dots\ f_N]$ in order to produce the required displacement. Thus the relationship between thruster configuration matrix B and thrusters forces f can be expressed as:

$$f = B^{-1}\tau \tag{28}$$

where the controller request τ derives from the equation of motion (Fossen, 1994). For a generic underwater vehicle one can write:

$$M\dot{v} + C(v)v + D(v)v + g(\eta) = \tau \tag{29}$$

$$\dot{\eta} = J(\eta)v \tag{30}$$

where M is the inertia matrix of the underwater vehicle, including rigid-body and added mass effects $M = M_{RB} + M_A$, C is the matrix of centrifugal and Coriolis terms $C = C_{RB} + C_A$, D is the matrix of damping terms, g is the vector of restoring forces (gravity and buoyancy) and $v = [u\ v\ w\ p\ q\ r]$ is the vector of linear and angular velocities in vehicle's body-fixed frame. Thus the generalized force vector τ represents the forces acting on the underwater vehicle with respect to the vehicle's body-fixed frame.

In our implementation the thrust efficiency η_T is introduced in the solution of allocation problems (28) by modifying the matrix B^{-1} weighting the contribution of individual thrusters according to their estimated efficiencies. This enables the vehicle to redistribute thrust on healthy thrusters (Hanai et al, 2010) and continuing with normal navigation.

3.4.1 Thruster Remapping

This approach, known also as *weighted generalized inverse*, makes use of a weight matrix W where the coefficients ($w_i = \eta_{T_i}$ and $0 \leq w_i \leq 1$) on the diagonal represent the estimated thrust efficiency for the platform's thrusters.

$$W_{(N\times N)} = \begin{bmatrix} w_0 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & w_N \end{bmatrix} \tag{31}$$

In normal conditions each coefficient can be assumed equal to 1, meaning that the specific thruster is healthy and fully available for producing the requested thrust. In the case of

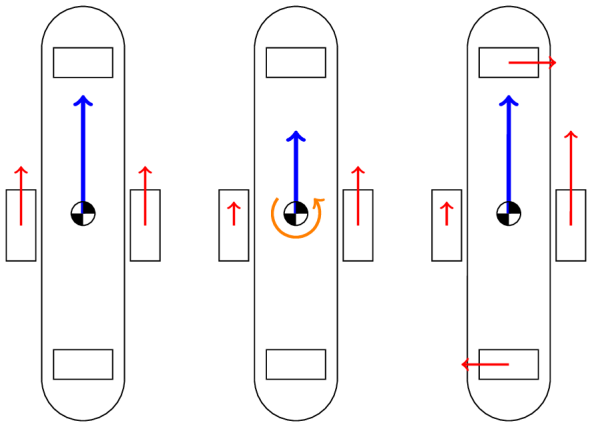


Fig. 7: Comparison of thrust allocation in Nessie VIII AUV. The diagram on left shows the reference case where a given amount of force has been requested along the surge axis. The middle case shows the effect of the degradation of the port-side actuator without any compensation. An extra torque is produced by the unbalance between thrusters and the total force request is not met. The right shows the effect of the thruster allocation algorithm where other actuators are used to compensate for the faulty thruster.

a complete failure $w_i = 0$ the use of this approach guarantees that the faulty actuator is excluded from the solution of the allocation problem. With the introduction of the weight matrix (31) the (28) can be rewritten as:

$$f = B_w^{-1}\tau \tag{32}$$

where the B_w^{-1} is known as weighted generalized pseudo-inverse of the thrust configuration matrix:

$$B_w^{-1} = W^{-1}B^T\left(BW^{-1}B^T\right)^{-1} \tag{33}$$

The use of a generalized inverse (Fossen, 1994; Omerdic and Roberts, 2004) avoids the presence of singularities (e.g. when thrusters are excluded from the solution $w_i = 0$) and is general enough to deal with non-square thruster configuration matrices (e.g. when thrusters are greater than the controlled degrees of freedom). It can be efficiently computed with the use of singular value decomposition (SVD) (Strang, 2006). An example of thruster remapping is shown in Fig. 7 where a normal allocation is compared to a faulty behaviour. In this case the port-side forward thruster is affected by degradation. Under these conditions an additional torque is generated by the unbalanced operation of forward thrusters. By remapping the use of actuators the remaining thrusters are used to compensate this unwanted behaviour and if allowed the starboard-side thruster increases its effort to match the original force request.

3.4.2 Thruster Saturation

Despite its flexibility the introduced approach cannot guarantee the allocated force vector f to stay within the saturation limit of each thruster. In fact the presence of a degradation can require extra effort from the remaining actuators in order to compensate for the lack of thrust. In some cases, however, this can lead to saturation of other thrusters which under normal conditions are guaranteed to stay within their limits. In order to solve this problem a solution has been proposed (Sarkar et al, 2002) but it requires modifications of the controller architecture by introducing extra integrators at the input channels of the underlying controller system. In developing this work we decided to leave the controller unmodified and to follow a different approach. This does not increase the complexity in lower level controllers, allowing to reuse existing modules, while, at the same time, preventing the effect of saturations by optimizing the force allocation policy. A similar method has been explored in previous works (Omerdic and Roberts, 2004; Johansen et al, 2004) and relies on the use of quadratic programming (QP) to iteratively derives a solution for the allocation problem. Other techniques have been also used in the past, such as S-approximation and T-approximation, but they are focused on mitigating the effect of force unbalance rather then avoid the saturation problem. These are not discussed in this work.

Our implementation relies on a customized version of the original QP formulation, with the use of CVXPY (Diamond and Boyd, 2016), a Python-embedded modelling language for convex optimization problems, and the state-of-the-art ECOS (Domahidi et al, 2013) solver. These provide a efficient way of implementing a real-time optimization method that has been integrated in the software architecture of an existing AUV.

The quadratic optimization problem (QP) is defined as following:

$$\min_{f,s} f^T P f + s^T Q s + \beta \bar{f} + \xi \tag{34}$$

subject to:

$$s = \tau_d - B\,f \tag{35}$$

$$-f_{act} \leq f \leq f_{act} \tag{36}$$

$$-\Delta f \leq f - f_{pre} \leq \Delta f \tag{37}$$

where the first term $f^T P f$ takes into account the power consumption of actuators, through the use of a matrix P, the second $s^T Q s$ penalizes the error s between request τ_d and achieved generalized force $B_w f$. The slack variable s is required to allow always a feasible solution to be found in the optimization problem. The matrix Q is chosen as $Q \gg P > 0$ so that solutions are found with $s \approx 0$. The matrix P is built approximating the thruster characteristic as a quadratic function: $h(f_i) = af_i^2 + bf_i + c_i$. Moreover the term f_{act} is the

available thrust for a given actuator and $\bar{f}$ is the maximum component of the f vector. These are defined as:

$$f_{act_i} = \eta_{T_i} f_{max_i} \qquad i \in \{0,\ 1,\ \ldots,\ N\} \tag{38}$$

$$\bar{f} = \max_i |f_i| \qquad \bar{f} > 0 \tag{39}$$

where f_{max_i} is the maximum thrust for the i-th thruster. These quantities are used in the above formulation to bound the actuator forces using the estimated thrust efficiency (36) and to penalize solutions where a single thruster is working at its maximum capacity (34) by introducing a coefficient $\beta > 0$. At the same time, actuator dynamics are enforced using (37), where f_{pre} is the solution found during the previous step and Δf represents the maximum rate of change for the actuators' output. The use of such constraint allows to discard solutions that abruptly modify the throttle commands. Lastly, ξ is a relaxation term that groups coefficients needed to avoid singularities and numerical errors.

Furthermore to better enforce a stable navigation in presence of different failures a *force prioritization policy* has been also implemented. This allows the vehicle to use the remaining available thrust in an ordered way, satisfying first force requests for high priority degrees of freedom (DOFs). An example of a possible policy ω for an inspection task is $heave > pitch > yaw > sway > surge > roll$. This privileges adjustments of depth and orientation to maintain the alignment of on-board sensors rather while relaxing the control on forward and lateral displacements. This is implemented by adjusting the coefficients q_i of the diagonal matrix Q such as:

$$q_i > q_j \quad \text{if } \omega(i) > \omega(j) \qquad i \neq j \wedge i, j \in \{0, 1, \ldots, N\} \tag{40}$$

still under the condition $Q \gg P > 0$. Such approach makes the solver prefer solutions where the error term s is lower for higher priority DOFs.

3.4.3 Speed Constraints

Beside the saturation prevention the mitigation algorithm also adjusts *cruise speed* for the vehicle in case a relevant failure is detected. This is to keep the controller's requests as close as possible to the actual thrusters' limits and to limit the compensation of unbalanced forces. Such procedure, beside estimating a numerical value, allows the fault mitigation node to share its knowledge with other autonomy modules using a range of preferred navigation speeds along the principal axis of the AUV. The updated speed is calculated by estimating the available force on each DOF. This is derived from the remaining force of each thruster f_{act_i}, the estimated efficiencies η_{T_i} and the thruster configuration matrix B, which provides the relationship between actuators and their contribution to each degree of freedom. In other words the available generalized force for a specific DOF can be written as:

$$\tau_{avail}(d) = \sum_i f_{act_i} = \sum_i \eta_{T_i} f_{max_i} \quad \text{with} \quad i \in T(d) \tag{41}$$

where $T(d)$ is the set of thrusters that are acting on the given degree of freedom d. After calculating (41) for all the controlled degrees of freedom the vehicle navigation speed can be expressed as:

$$v_{allowed} = v_{cruise} \sqrt{\frac{\tau_{avail}}{\tau_{nominal}}} \qquad 0 \leq v_{allowed} \leq v_{cruise} \tag{42}$$

where $v_{allowed} = [u\ v\ w\ p\ q\ r]$ is the adjusted cruise speed vector, v_{cruise} is the initial speed set by the planner system, under the constraint $v_{cruise} \leq v_{max}$ where v_{max} is the maximum speed the vehicle can achieve, and $\tau_{nominal}$ is the force available in normal conditions without any fault.

The use of this expression is explained by physical effects of individual terms of the equation (29) on the motion of the vehicle. In particular for underwater vehicles, the damping term D is often approximated with the only use of *quadratic drag* coefficients (Fossen, 1994). Such simplification assumes the drag force proportional to the square root of the speed. In the AUV case the drag force represents a strong contribution to the set of forces acting against the motion of the vehicle. Therefore by reducing the navigation speed the force requested from controller during cruise phases decreases accordingly.

3.5 Knowledge Representation

The fifth module is the *knowledge base* high level components. This collects the output of *fault mitigation* submodules and the estimations done at *fault detection* level to evaluate the effect of detected failure on the vehicle's effectiveness using different levels of abstraction. This module offers its evaluations to planner modules that, if needed, can adapt, restructure or trim the sequence of tasks that compose a long-term autonomous mission. An overview of this module is shown in Fig. 8 where the example of an inspection mission for a hover-capable AUV is presented. The proposed schema is a layered structure which connects and organises the vehicle's internal belief at different abstractions levels thus isolating basic and measurable knowledge elements from complex features.

Concepts at the *components* level represent a physical device, an actuator or a sensor and its internal characteristics, like its health status or its constraints, which combined together give the vehicle a specific capability that can be used while executing a given mission. Concepts at the *capabilities* level describe the effect of the component's availability on the vehicle's characteristics. For instance at this

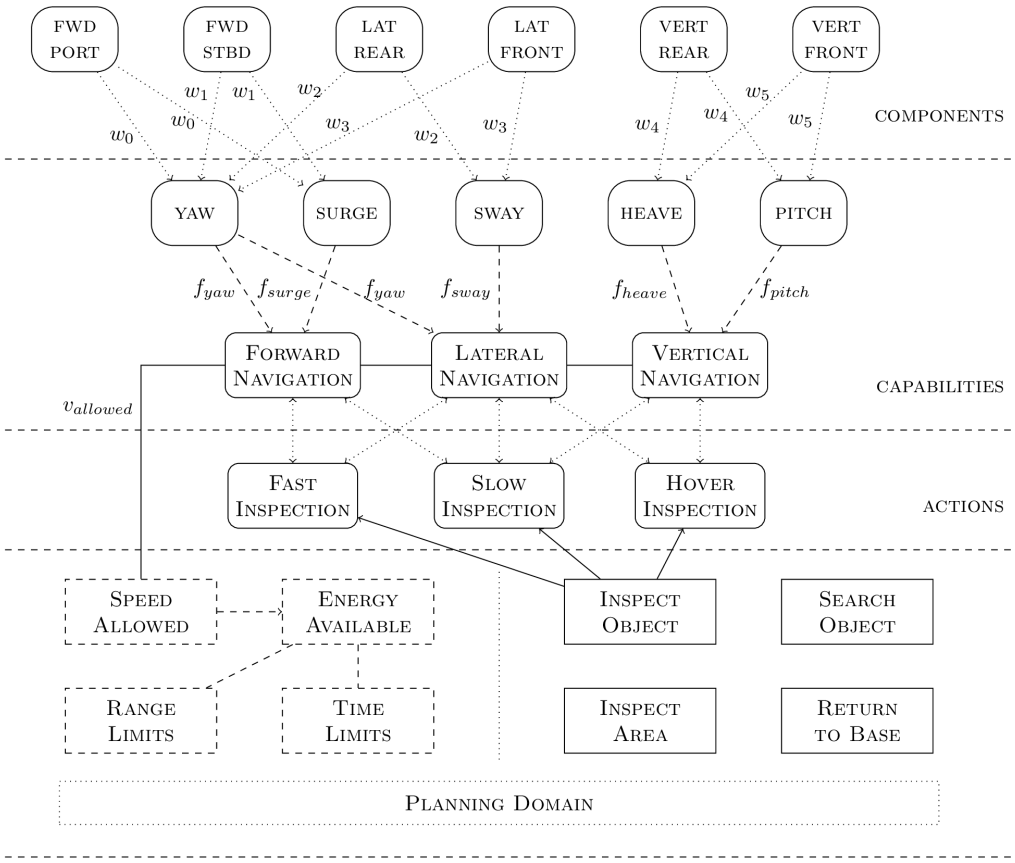


Fig. 8: Knowledge representation of the vehicle's status inside the *knowledge base* module. This software node can be queried for high-level concepts by other autonomy modules, providing a qualitative information about the current vehicle's capabilities and the available actions. Here is shown the relationship between an *object inspection* task and the coefficients estimated by the *fault mitigation* architecture in terms of vehicle's *components*, *capabilities* and *actions* abstractions.

level the navigation capabilities are represented as a set of logical relationships among the control forces, control laws and navigation sensors. Concepts at the *actions* level, instead, represent the highest level of abstraction from the vehicle's *components* and *capabilities*. These elementary actions make use of the underlying concepts in order to achieve a unit of work, like moving the vehicle along a trajectory or using a combination of sensors to scan a portion of the seabed. Actions can be seen also as the basic building blocks that compose a more complex item, the *task*. This is a specific sequence of actions that produces a change in the mission's state. End users can often encode tasks in the vehicle by using a scripting language or a list of basic commands which will govern the behaviour of the vehicle during the duration of the task. The concept of the task is then made available in the domain of the *planner* allowing the autonomous operation of the vehicle if requested by the mission constraints.

The use of these abstractions allows the *knowledge base* node to connect the health status of the vehicle's components to their effect on its capabilities and actions. In Fig. 8 we show an example of this interaction for the operating domain of the proposed fault mitigation system. In this case the *planner* has to execute an inspection mission by a sequence of different *inspection object* tasks. In this example for each task the planner can reach its goal by selecting one of the available *actions* related to the task. Each action will presents to the planner a different *execution cost* derived from the analysis of the underlying status of the vehicle's *capabilities* and *components* and the characteristics of the action. In the event of *thruster degradation* the fault's knowledge is thus propagated from the *components* to *capabilities* layer using the estimates of the *diagnostic* system and later to the *actions* layer using their logical relationships with the required capabilities, updating the *weights* of each logical connection and thus influencing the behaviour of a planner module.

3.5.1 Energy Estimation

A relevant aspect of the *knowledge base* module is the update of energy usage estimations if a change in the vehicle's configuration is implemented. This sub-component has been introduced taking into account the availability of speed constraints, provided by lower level modules, in the event of failures. Its role is providing support for the planning system in adapting the mission with more precise energy usage assumptions that otherwise can produce a less effective mission plan. This is especially true in case of degradation failures, where a capability is not completely lost but affected by the fault. In such cases the mitigation framework can only correct the vehicle's motion, maintaining good navigation capabilities only for a reduced range of cruise speeds. Therefore the use of an up-to-date $v_{allowed}$ vector allows the vehicle's architecture to be better at estimating the operational navigation range and energy usage without any interaction with external users.

These estimations, often encoded at the planning level, can be performed according to simple usage models often found in the literature (Bradley et al, 2001; Willcox et al, 2001) about oceanic surveys with underwater vehicles. Considering the inspection of a large mission area the energy usage can be written, without considering environmental effects, as:

$$\varepsilon_{tot} = \left[\frac{\rho C_d S v_{surv}^3}{2\eta_p} + H \right] T_{surv} \tag{43}$$

where v_{surv} represents the survey speed, T_{surv} the survey duration, η_p the propulsion efficiency, C_d the drag coefficient, ρ the density of the water, S the vehicle's surface area and H the platform's hotel load. The use of this equation allows the planning system to correlate the energy needs of the inspection task with two mission parameters v_{surv} and T_{surv} often taken into account in the planning domain. This aspect, together with the limited availability of resources in the vehicle, identify an important set of constraints that the planner needs to take into account when adapting the mission at runtime.

On the other hand the use of the expression (43) shows that the energy usage can be minimized by selecting the appropriate trade-off among these quantities. In fact assuming $v_{surv} \approx L/T_{surv}$ and dividing the (43) by L the total survey distance:

$$\frac{\varepsilon_{tot}}{L} = \frac{\rho C_d S v_{surv}^2}{2\eta_p} + \frac{H}{V} \tag{44}$$

gives the expression of the energy use per unit of distance. Deriving this quantity with respect to the survey speed v_{surv} and setting it to zero gives estimate of the optimal speed:

$$v_{opt} = \sqrt[3]{\frac{H\eta_p}{\rho C_d S}} \tag{45}$$

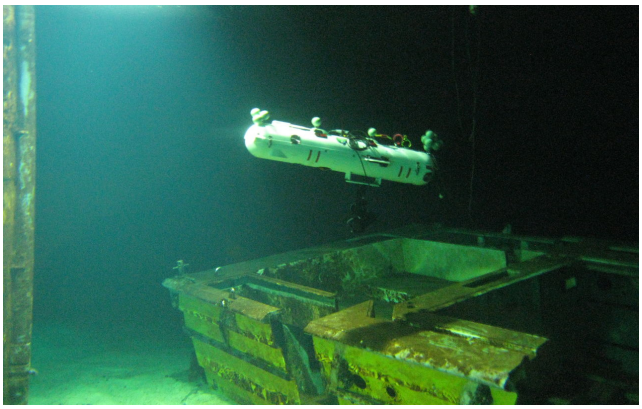


Fig. 9: Nessie VIII AUV while conducting an inspection mission in an artificial indoor environment. In this scenario the vehicle is inspecting a wrecked ship near a man-made structure represented by a set of pillars and pipes.

The use of (45) in the (43) allows thus the calculation of the minimum energy requirement for a given inspection task:

$$\varepsilon_{min} = \frac{3LH}{2v_{opt}} = \frac{3}{2} T_{surv} H \tag{46}$$

This expression, while representing the lower-bound of the energy usage, can be used during the initial stages of the mission planning where reduced knowledge about the environment and ideal conditions are often assumed.

4 Experimental Results

In order to evaluate the performance of the proposed system a series of experiments has been conducted using a real platform Nessie VIII AUV. First the operational experience collected during sea trials is reported. The vehicle has been deployed in a real shallow water environment in the context of PANDORA project's sea trials. The main goal is testing the performance of the architecture's components in presence of sea currents, real disturbances and longer execution times. Second a series of detailed tests have been conducted for evaluating the behaviour of the proposed architecture both in controlled environment and during real sea operations. The vehicle is assigned with a set of inspection tasks to be executed with given constraints on execution time and energy consumption. Concurrently the effectiveness of the fault mitigation system and its individual components is evaluated under the presence of faults.

4.1 AUV Platform: Nessie VIII

The underwater vehicle used in this work is based on the updated version of the Nessie AUV (Valeyrie et al, 2010)

Fig. 10: Satellite view of the mission area. The pier structure stretches from the Fort William's shore into the *Loch Linnhe* waters. In this satellite view the mission area (yellow) and the restricted area (red) have been marked.

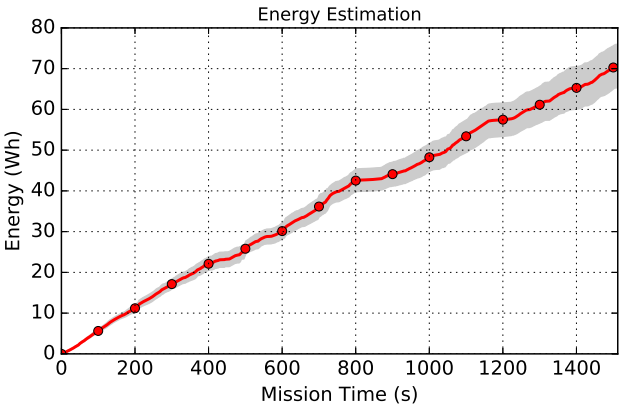


Fig. 11: Energy usage estimation during the execution of a long inspection task (*day4-1*) in real sea conditions in presence of strong tidal currents.

platform, shown in Fig. 9. This vehicle is a torpedo-shaped AUV with hovering capabilities. It has a 0.28 m diameter, a length of 1.75 m, it weights almost 60 kg and it has a maximum operation depth of 70 m. This platform includes different navigation sensors like fiber optic gyroscope, GPS unit, DVL and pressure sensor. The vehicle is powered by a set of four high-energy lithium polymer (LiPo) batteries, each of them rated for 543.9 Wh, 25.9 V and 21 Ah. This give the vehicle a total available energy of about 2.2 kWh. Enough for long running operations in a harsh environment with external payload without the need of interrupting the mission for recharging. The propulsion is given by six Seabotix HPDC1502 DC thrusters, each capable of producing about 250 W of power and 4.5 kg of thrust in ideal conditions. The thrusters are aligned with the principal axis of the platform, two for the longitudinal, two for the lateral and two for the vertical axes. This configuration has been chosen as a trade-off between operational requirements and design constraints. An energy measurement system has been integrated with the vehicle. It relies on the use of several hall-effect current sensors (ACS715) to keep track of the usage of energy at runtime. This task is off-loaded to a microcontroller unit made independent from the main control software. The choice of such sensors has been dictated by the use of a single 28 V power bus and high current (greater than 10 A) connections in the AUV design.

4.2 Operational Experience

The PANDORA project's sea trials experience gave the opportunity of validating the proposed architecture in the context of a real autonomous mission. The vehicle is assigned several tasks to be executed in a partially known scenario. The experimental site is the Underwater Centre's facilities, in Fort William, Scotland. There were five days of operations. This environment is characterized by the presence of a marine pier, which can be seen in Fig. 10, that extends in the *Loch Linnhe* waters, a sea inlet in the north-western Scotland. This inlet extends for about 50 km south-west to the *Firth of Lorn* running to the top of the *Great Glen Fault*. Despite the relative dimensions this area is characterized by the presence of strong tidal currents with tides low enough to make part of the sea floor around the pier emerge during their lowest period. The operational area involves the pier structure and its surrounding water giving the vehicle approximately 0.1 km^2 of navigation area. The presence of other activities (mostly diving and ROV training) at the operational site during the trials required us to define some restricted zones where the autonomous operations were not allowed as can be seen in Fig. 10. This area is characterized by the presence of high tides which can affect change the operational environment multiple times during a single day. Trails have been carried out during the spring tides time in order to evaluate the AUV platform when operating in harsh conditions and with strong tidal currents while remaining in shallow waters. Concurrently with the project's trials some of the architecture's main components have been evaluated during the last 3 days of the measurement campaign.

First, the use of *thruster model* as an on-board energy estimator has been taken into account. In this case the energy usage of the modelled thrusters ε_m has been recorded during these experiments and compared to the real measurements ε_r at the end of each mission. Results are shown in Table 1 where experiments on the same day have been conducted without recovering the vehicle.

$$NRMSE(\varepsilon_m) = \frac{RMSE(\varepsilon_m)}{\varepsilon_{r_{max}}} \tag{47}$$

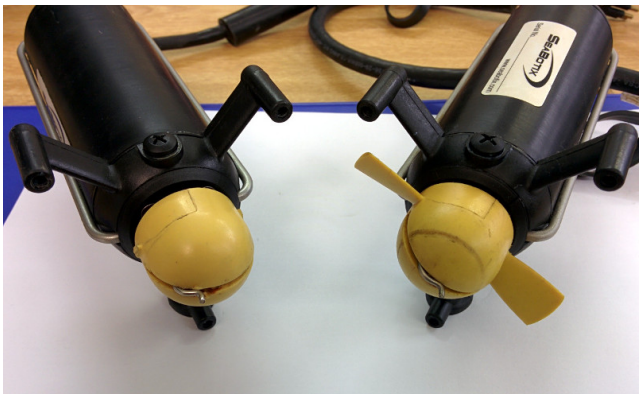


Fig. 12: Detail of the Nessie AUV lateral thrusters. On the left the broken actuator where both of propeller's blades are missing, due to a possible ingestion of sea weed inside the thruster's nozzle.

The normalized root-mean-square error (NRMSE) is used to provide a qualitative metric to evaluate the estimation done in the vehicle while running the assigned tasks. In this case this is defined as the ratio between the root-mean-square error (RMSE) and the maximum value for the short-term energy $\varepsilon_{r_{max}}$. The RMSE is thus written as:

$$RMSE(\varepsilon_m) = \sqrt{\frac{\sum_k^{N_s}(\varepsilon_m[k] - \varepsilon_r[k])^2}{N_s}} \tag{48}$$

where ε_m values are the model predictions, ε_r values the real measurements and N_s is the number of samples used to calculate this metric. By considering the results shown in Table 1 and the trend in Fig. 11 we can observe that the use of the *thruster model* as energy estimator allows the vehicle to track its energy usage with an error of 10% when executing missions up to about 1 hour and 1 kilometre of navigation around marine structures. This provided us a good confidence in using the data-driven model in a simulated environment for evaluating the performances of Nessie VIII AUV and better planning outdoor missions. Furthermore these results show that energy usage metrics are good candidates for evaluating the behaviour of the vehicle at runtime. They can provide a qualitative indicator of the vehicle's status not only from a diagnostic point of view but also about its performance across a longer time window.

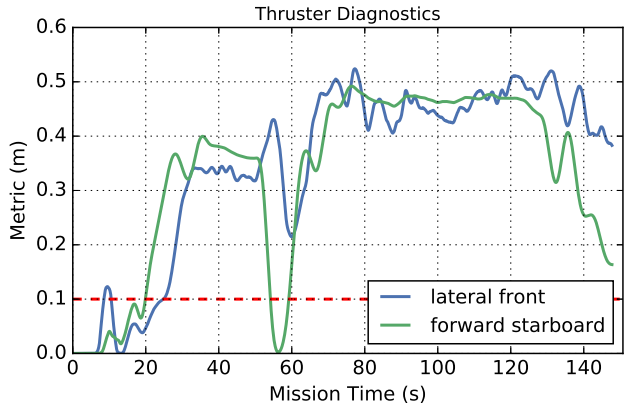
Table 1: Results of inspection experiments done during the sea trial campaign with Nessie VIII AUV navigating around the marine structure in Fort William.

Dataset	Dur. (s)	Dist. (m)	Energy (Wh)	NRMSE (%)
day3-1	2040	246	63.74	7.28
day3-2	608	104	23.42	7.49
day3-3	1156	186	28.09	8.99
day3-4	2417	709	143.77	8.44
day4-1	3432	1258	197.46	10.46
day5-1	2440	736	130.03	2.84
day5-2	890	433	47.15	2.77
day5-3	1513	512	67.07	2.91

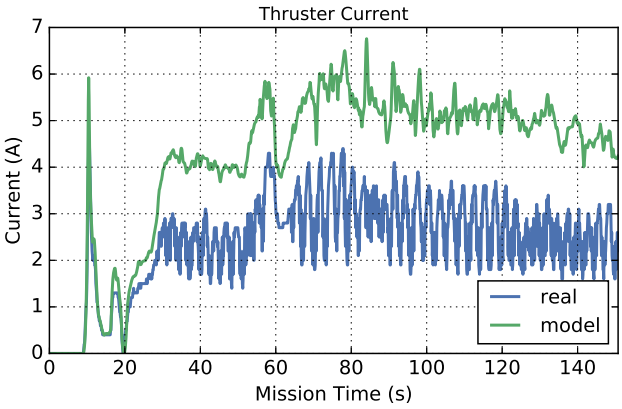
Second, the *fault model* and the robustness of the *diagnostic system* has been tested. During the last day of the sea trial campaign Nessie AUV experienced a real failure in one of its thrusters. The fault, discovered at the end of the day, affected the *lateral front* thruster with the loss of the propeller's blades. This is possibly due to ingestion of sea weed in the thruster's tunnel, causing the effect shown in Fig. 12. The *diagnostic system* was able to track the failure for all the remaining part of the daily operations, including some experiments with the injection of other faults. This provides a way of comparing the behaviour, in terms of diagnostic metric, of both a real failure and an injected one. For this specific type of fault the damaged component shows a trend similar to an injected fault with a *high degradation* profile ($\eta_T = 0.4$), as can be also seen in Fig. 13a. This can be explained by observing how the current usage pattern for the *broken* thruster deviates from its predicted trend by the *thruster model*, as shown in Fig. 13b. Beside the numerical difference the damaged actuator also presents a highly variable trend possibly caused by the effect of the thruster's internal speed controller and stall protection. The failure experienced in the real operations motivates the use of a *tracking algorithm* for the *thruster efficiency* (η_T) parameter within the proposed system. This is because in presence of complex failure scenarios, like a *failed propeller*, the system is able to rely on the *short-term energy* feature to reduce the estimated efficiency even without introducing a dedicated estimation procedure[2]. In this case the efficiency parameter is adjusted until a mismatch is found between the nominal characteristic and the measured feature and a lower limit is reached thus removing it completely from the *thrust allocation* problem. In such a case the *knowledge base* is notified about this exclusion and its internal belief about the degrees of freedom's availability is updated.

The presence of a real fault is also shown analysing the difference between model estimation and real energy usage for the faulty thruster, as shown in Fig. 14. In this case despite the residual uncertainty in the model, the real usage falls below the model estimation. This is due to the fact the actuator is working outside its nominal range making the presence of this kind of failure detectable even with a qualitative analysis on such trends.

[2] In an ideal scenario this requires the vehicle to interrupt its current task, execute a specific set of actions and compute an accurate estimation of the *thrust efficiency* (η_t) increasing even more the mission duration. Such a procedure should be trigged by a planning framework.



(a) Comparison of filtered diagnostic metric during a navigation task between an injected fault on the *forward starboard* thruster ($\eta_T = 0.4$) and a real failure in *lateral front* thruster (*broken propeller*).



(b) Analysis of current usage pattern for the *lateral front* failed thruster. The plot shows the difference between the real measurements and the expected behaviour according to the relative thruster model.

Fig. 13: Analysis of real failure for the *lateral front* thruster of Nessie VIII AUV during sea trials.

4.3 Fault Analysis

Together with the operational experience further testing has been conducted to evaluate the performance of the proposed architecture under the effect of faults. With this in mind more experiments are conducted in different scenarios: an indoor test tank in steady environmental conditions and in real sea environment, again, in the context of the PANDORA's project field trials. They benchmark the proposed architecture by injecting different levels of failures. These affect the *forward* and *lateral* thrusters of Nessie VIII AUV. These actuators are used for controlling the *surge*, *sway* and *yaw* degrees of freedom. While running such experiments the execution time and the energy usage of each inspection task is measured. This enables to compare the effects of the introduced algorithms with the normal behaviour of the platform when no failure is present. To better represent the severity of the injected faults the fault classification, shown in Table 2, is introduced. Six different fault levels are taken into account. As described in Section 3.3.1 different modes of thruster degradation can be covered by the proposed system. In this first set of experiments the maximum available thrust has been limited according to injected level of degradation. This assumes a change in the thrust characteristic similar to the one shown in Fig. 6 in the case of the *low degradation* fault.

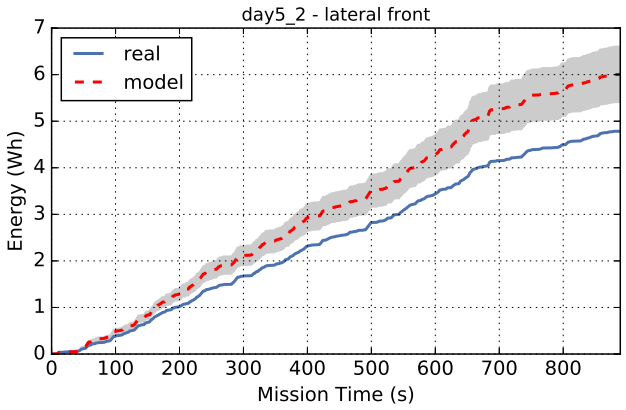


Fig. 14: Energy usage estimation for a faulty thruster during long inspection task (*day5-2*). In this case the real energy usage falls below what estimated by the model. This confirms that the *lateral front* thruster is not operating correctly.

4.3.1 Controlled Environment

The first set of experiments takes place in a 12m x 9m indoor tank, shown in Fig. 15, with steady environmental conditions across different runs. The vehicle is given the task of inspecting the corners of the pool looking for the presence of submerged objects. This task is executed several times in presence of the different fault configurations, as described

Table 2: Thruster degradation levels used in the experiments with Nessie VIII AUV. The thrust values are calculated with respect to their nominal characteristic.

η_T	Max. Thrust (N)	Level Description
1.0	39.24	*nominal conditions*
0.8	31.39	*low degradation*
0.6	23.54	*medium degradation*
0.4	15.70	*high degradation*
0.2	7.85	*severe degradation*
0.0	0.00	*complete loss of the actuator*

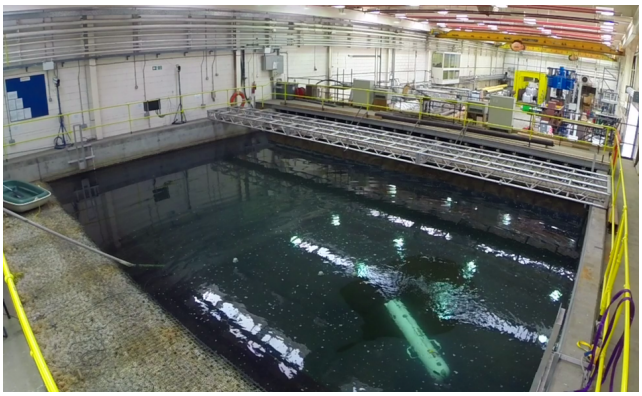


Fig. 15: The Nessie VIII AUV deployed in the indoor tank before starting the experiments. The vehicle is hovering at fixed depth before resuming its inspection task and navigate along the first trajectory leg where faults are injected.

earlier in Table 2, introducing, in the first instance, the failure in a *forward* thruster and later in a *lateral* one. In the case of Nessie VIII AUV the *forward* thrusters are mostly used for the *surge* degree of freedom with a small contribution to the *yaw* only when navigating above manoeuvring speeds while the *lateral* ones are mainly used for the *sway* and *yaw* degrees of freedom especially when navigating at lower speeds, for instance during the hovering phase of an inspection task. Vertical thrusters are not consider in those experiments as the lack of redundancy on the *heave* DOF prevents this vehicle to maintain efficiently in attitude during navigation.

Under these considerations Fig. 16 and Fig. 17 show how thruster degradation affects trajectory tracking capabilities of Nessie VIII AUV. Deviations from planned trajectories are explained with the presence of an additional torque generated by the unbalanced thrust allocation among forward actuators. In these charts the vehicle's navigation is reported for both nominal and faulty scenarios. The framework detects a mismatch between actuators in the first part of trajectory (i.e. $\overline{AB}$ segment) and adjusts progressively the allocation coefficients to overcome the detected failure. After reducing the estimated efficiency of any degraded thruster the navigation resumes like in the nominal scenario. The right-hand side of Fig. 16 shows, in detail, the effect of an uncompensated failure in the same scenario. The use of the *mitigation framework* allows the vehicle to recover and correct its navigation while executing its tasks. Numerical results are presented in Table 3 and 4, respectively, for the *forward degradation* and *lateral degradation* scenarios. The tables show how different degradation levels affect the execution of the inspection task. In both cases the vehicle managed to complete the assigned task while they differ in terms of performance under the presence of *medium* to *severe* degradations.

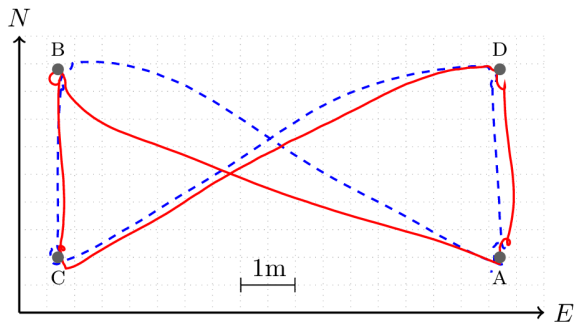
In the case of the Nessie VIII AUV an increase of 60% and of 100% in the use of *time* and *energy* resources during the execution of a single tasks are considered as *warning* and *critical* thresholds. These are used in the *mission execution* module to trigger the adaptation of the mission. This, for instance, may lead to the abort of the current task if the resources needs to be allocated on a more important task. Concurrently a *replanning* procedure is usually requested taking into account the updated state of the platform. One key aspect that needs to be considered when looking at these results is that the goal of the *fault mitigation algorithm* is to strictly provide an appropriate allocation strategy that enables the vehicle to continue its current task. This, as shown by these results, is done at the expense of two mission resources, available energy and remaining time. These, while of lesser importance from a pure control point of view, play an important role in supporting the calculations done at the *planning* system level when calculating the next sequence of actions. With this in mind the acceptable usage range, in terms of performance degradation, for the proposed architecture can be identified for the underwater vehicle used in these tests. The proposed approach allows the platform to survive up to a 60% loss or *high degradation* ($\eta_T = 0.4$) in its *forward* actuators and up to 40% loss or *medium degradation* ($\eta_T = 0.6$) in its *lateral* ones without disrupting the assumptions usually taken at the planning level. This behaviour matches what was initially observed in simulation

Table 3: Experimental results for inspection task in controlled environment in presence of *port-side forward thruster degradation*. The vehicle completed the inspection task with success for all the different levels of degradation.

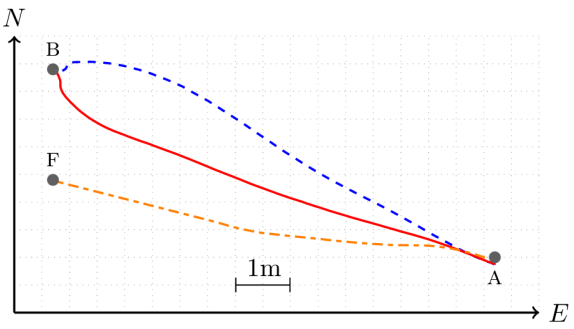
η_T	duration (s)	energy (Wh)	rel. dur. (%)	rel. ene. (%)
1.0	126.11	3.60	–	–
0.8	140.99	4.19	11.80	16.30
0.6	148.92	4.62	18.08	28.37
0.4	176.68	5.80	40.09	61.03
0.2	203.77	6.96	61.57	93.11
0.0	204.31	7.02	62.00	94.94

Table 4: Experimental results for inspection task in controlled environment in presence of *front lateral thruster degradation*. For $\eta_T > 0.6$ vehicle's efficiency is severely reduced and the use of resources is sensibly increased.

η_T	duration (s)	energy (Wh)	rel. dur. (%)	rel. ene. (%)
1.0	126.11	3.60	–	–
0.8	137.68	4.16	9.18	15.52
0.6	137.30	4.68	8.88	29.87
0.4	197.36	8.03	56.50	123.14
0.2	203.27	8.49	61.18	135.75
0.0	205.95	8.62	63.31	139.37

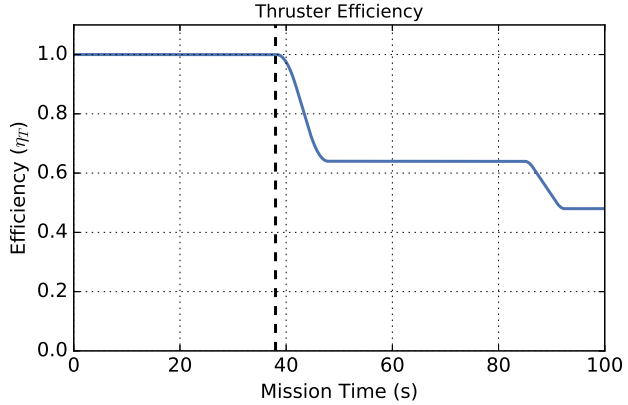


(a) Dashed blue line shows the navigation path under nominal conditions. Solid red line shows the path in presence of fault condition using the proposed system. Along the first $\overline{AB}$ segment the framework adapts the underlying control system resuming a normal navigation for subsequent segments $\overline{BC}, \overline{CD}, \overline{DA}$. Navigation is point-to-point with 360-degree rotations around points A, B, C, D to acquire sonar images.

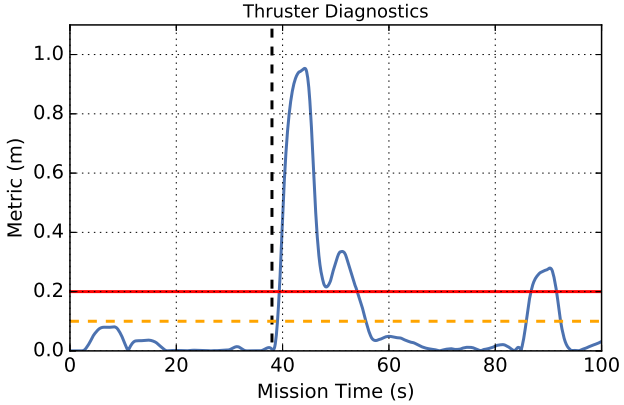


(b) Dashed blue line shows the navigation path under nominal conditions. Solid red line shows the path using the mitigation algorithm during initial adaptation (e.g. adjustment of w_i coefficients). Double dashed orange line shows the path without mitigation. In this case the vehicle fails to reach the assigned waypoint B before aborting its navigation once reaching point F.

Fig. 16: Trajectory comparison for controlled environment experiments with *forward thruster degradation* faults ($\eta_T = 0.2$).



(a) Trend of the estimated thruster efficiency (η_T) during a *forward degradation* experiment. After injecting the fault ($T = 38.0s$) the *mitigation algorithm* continuously estimates the efficiencies while adjusting the *weighting coefficients* (w_i) to mitigate the effect of the degradation.



(b) Trend of the diagnostic metric during a *forward degradation* experiment. The two horizontal lines represent the threshold used in the fault detection algorithm. These prevent false alarms from triggering the mitigation algorithm while limiting its responsiveness.

Fig. 17: Experimental results for an inspection task in presence of *high degradation* ($\eta_T = 0.40$) on the *port-side forward thruster* of Nessie VIII AUV. The mitigation architecture estimates a thrust efficiency close to the injected one ($\eta_{T_{est}} = 0.46$) due to the use of wide thresholds but still allowing the vehicle to complete its mission without disrupting its navigation.

studies. Furthermore this suggests that for *severe degradation* faults an additional mitigation strategy is needed to achieve reasonable performances in the context of a long running mission.

4.3.2 Real Sea Conditions

Together with the controlled conditions further testing has been conducted with the context of the PANDORA's project sea trials. A set of navigation tasks similar to what has been tested in the simulated and controlled environments has been assigned to the vehicle operating next to a human-made structure in shallow waters, as shown in Fig. 10 and 18. In this case additional constraints have been applied to the mission area and the vehicle's trajectory in order to control the effects of tidal currents and allow safe operations. The results of these fault analysis experiments are shown in Table 5 and they are close to what have been shown for the controlled environment. Also for the real sea operations the mitigation system has been able to provide assistance in case of *medium-high degradations*. In the case of more *severe* fail-

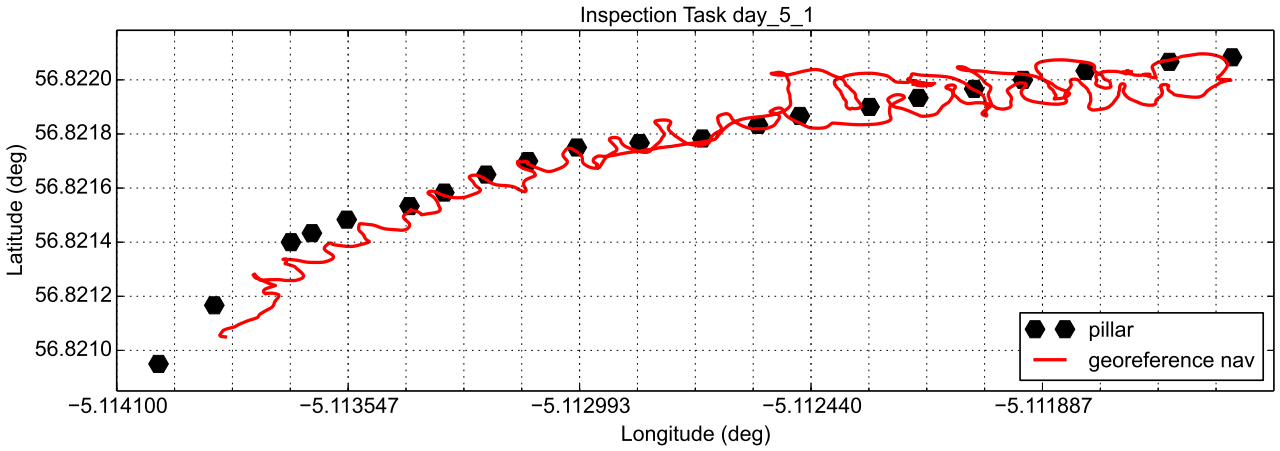


Fig. 18: Uncompensated navigation data for a long running inspection task. The vehicle uses its GPS unit to acquire the initial position before diving and starting the inspection task. The injection of *low* and *medium* faults does not prevent the vehicle to continue its current task even though its navigation accuracy is affected (as shown in top right part of this chart).

ures the vehicle wasn't able to keep up with the rising tidal currents up to a point when, after the injection of the *complete loss* failure, the experiments have been suspended due to the vehicle drifting considerably far from the assigned trajectory. These experiments highlight the differences between controlled environments and sea operations, where the presence of currents, wind and waves can affect by great amount the performance of the vehicle if severe failures are experienced.

Table 5: Experimental results for navigation tasks in presence of tidal currents. The case for $\eta_T = 0.0$ resulted in a mission abort to due violation of trajectory tracking constraints.

η_T	duration (s)	energy (Wh)	rel. dur. (%)	rel. ene. (%)
1.0	80.96	4.29	–	–
0.8	85.52	4.62	5.63	7.72
0.6	88.01	5.81	8.71	35.54
0.4	134.41	7.32	66.02	70.81
0.2	139.24	8.82	71.99	105.72
0.0	–	–	–	–

5 Discussion

The advantage of the proposed approach lies in the use of energy consumption measurements to assess the status of the vehicle and to drive an automatic fault mitigation system. Energy consumption is often already available in the vehicle's architecture for monitoring purposes (i.e. battery management systems, etc.). The choice of analysing a quantity already available at runtime helps to reduce complexity of the system design. This allows the proposed architecture to be implemented on several autonomous underwater vehicles without requiring big modifications but simply the addition of some sensors if an energy metering facility is not present.

The experimental results provide an insight on the performance of the proposed architecture. They show that the use of a data-driven approach for extracting the thruster model allows quick characterisation of the behaviour of these components. This is done with a good accuracy and without the need for dedicated test equipment other than the vehicle itself. Furthermore the use of the extracted model as an energy usage estimator has been validated in real sea conditions. This offers satisfactory estimation confidence for tasks lasting up to one hour even in presence of tidal currents. Such phenomena affect the system with the presence of strong external disturbances that can reduce the accuracy of the short-term fault detection in presence of *low degradation* faults. This is due to the use of adjusted detection threshold to reject an increased disturbance level and does not affect the system's capabilities of detecting more severe faults. On the other hand by looking at the long-term energy usage analysis the system is capable of qualitatively detecting deviations from the expected behaviour suggesting the presence of a possible failure even under the effect of strong external disturbances.

The effectiveness of the proposed energy-based diagnostic system is also analysed by comparing its behaviour under the effect of synthetic and real failures. The presence of a broken propeller is shown as a substantial change in the energy usage pattern of the damaged actuator. This, despite the presence of residual uncertainty and external disturbances, is detected and propagated to the output diagnostic metric in

a similar way of the synthetic faults. This validates the use of the proposed approach for handling different type of failures under a unified representation for the state of the faulty actuators.

The performance of the proposed automatic fault mitigation system (AFMS) is then analysed. The role of this system is to provide an automatic response to an unexpected event allowing the autonomous system to continue with its mission. The results shown in the previous sections are specific to the AUV used during the experiments and to the constraints of its assigned mission. On the other hand the experimental campaign suggests a procedure to evaluate the operational limits of such system by analysing the changes in energy usage and time efficiency in presence of incremental levels of performance degradation. This approach can be used with a generic AFMS and therefore characterize its usage for a long-term autonomy mission in presence of unknown environments.

6 Conclusion

The approach presented in this paper can be summarized as follows. A fault-mitigation architecture has been built around the concept of energy-awareness and its use is analysed with a hover-capable AUV in real sea conditions. First the characteristics of the vehicle's actuators are estimated using a data-driven approach; then the extracted model is used to drive a model-based diagnostic subsystem. This allows the detection of anomalies at runtime while tracking and monitoring the energy usage during the entire mission. This is done by comparing the modelled components with the sensors measurements over a long time scale. Upon the detection of faults the runtime estimates of *thrust efficiency* coefficients are used to adjust the lower-level control system and its force allocation strategy. This provides a first response to the detected anomalies allowing the vehicle to quickly react to unexpected events without interrupting its current task. Concurrently the information collected about the detected anomalies is propagated into a knowledge base. This allows high-level reasoning to provide a better awareness about the vehicle's internal state. This knowledge is then made available to planning modules that can re-adapt the vehicle's mission. This is shown in this work by propagating the allowed navigation speeds together with the availability of elementary actions.

The proposed architecture has been integrated in a real underwater vehicle with limited computational resources. Results have been shown in controlled and real sea conditions. In the latter case the presence of a real unplanned failure validated the system in presence of real disturbances and without relying on synthetic faults. We believe that the use of such architecture provides a good improvement in the autonomy capabilities of modern AUV platform and therefore we consider the energy-awareness as fundamental characteristic for preparing future vehicle designs to the persistent autonomy task.

Acknowledgements The authors would like to thank all the other members of the Ocean Systems Laboratory at Heriot-Watt University. The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 - Challenge 2 - Cognitive Systems, Interaction, Robotics - under grant agreement No 288273 - PANDORA.

References

Ahmadzadeh SR, Carrera A, Leonetti M, Kormushev P, Caldwell DG (2014a) Online discovery of auv control policies to overcome thruster failures. In: Proc. IEEE Intl Conf. on Robotics and Automation (ICRA 2014), Hong Kong, China

Ahmadzadeh SR, Kormushev P, Caldwell DG (2014b) Multi-objective reinforcement learning for auv thruster failure recovery. In: Proc. IEEE Symposium Series on Computational Intelligence (SSCI 2014), Florida, USA

Alessandri A, Caccia M, Veruggio G (1999) Fault detection of actuator faults in unmanned underwater vehicles. Control Engineering Practice pp 357–368

Antonelli G (2006) Underwater robots motion and force control of vehicle-manipulator systems. Springer

Antonelli G, Caccavale F, Sansone C, Villani L (2004) Fault diagnosis for auvs using support vector machines. In: Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on, vol 5, pp 4486–4491 Vol.5, DOI 10.1109/ROBOT.2004.1302424

Bradley A, Feezor M, Singh H, Sorrell F (2001) Power systems for autonomous underwater vehicles. Oceanic Engineering, IEEE Journal of 26(4):526–538, DOI 10.1109/48.972089

Caccia M, Bono R, Bruzzone G, Bruzzone G, Spirandelli E, Veruggio G (2001) Experiences on actuator fault detection, diagnosis and accomodation for rovs. International Symposiyum of Unmanned Untethered Submersible Technol

Cashmore M, Fox M, Larkworthy T, Long D, Magazzeni D (2014) AUV mission control via temporal planning. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp 6535–6541, DOI 10.1109/ICRA.2014.6907823

Corradini M, Monteriu A, Orlando G (2011) An actuator failure tolerant control scheme for an underwater remotely operated vehicle. Control Systems Technology, IEEE Transactions on 19(5):1036–1046, DOI 10.1109/TCST.2010.2060199

Cristofaro A, Johansen TA (2014) Fault tolerant control allocation using unknown input ob-

servers. Automatica 50(7):1891 – 1897, DOI http://dx.doi.org/10.1016/j.automatica.2014.05.007

De Carolis V, Lane D, Brown K (2014) Low-cost energy measurement and estimation for autonomous underwater vehicle. In: Proceedings of IEEE-MTS Oceans'14, Taipei, Taiwan

Dearden R, Ernits J (2013) Automated fault diagnosis for an autonomous underwater vehicle. Oceanic Engineering, IEEE Journal of 38(3):484–499, DOI 10.1109/JOE.2012.2227540

Diamond S, Boyd S (2016) CVXPY: A Python-embedded modeling language for convex optimization. Journal of Machine Learning Research

Domahidi A, Chu E, Boyd S (2013) ECOS: An SOCP solver for embedded systems. In: European Control Conference (ECC), pp 3071–3076

Fagogenis G, Flynn D, Lane D (2014) Improving underwater vehicle navigation state estimation using locally weighted projection regression. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp 6549–6554, DOI 10.1109/ICRA.2014.6907825

Fossen T (1994) Guidance and Control of Ocean Vehicles. Wiley, Chichester New York

Ge J, Roemer M, Vachtsevanos G (2004) An automated contingency management simulation environment for integrated health management and control. In: Aerospace Conference, 2004. Proceedings. 2004 IEEE, vol 6, pp 3725–3732 Vol.6, DOI 10.1109/AERO.2004.1368190

Hamilton K, Lane D, Taylor N, Brown K (2001) Fault diagnosis on autonomous robotic vehicles with recovery: an integrated heterogeneous-knowledge approach. In: Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, vol 4, pp 3232–3237 vol.4, DOI 10.1109/ROBOT.2001.933116

Hanai A, McLeod C, Rosa K (2008) A practical approach to the development of thruster models for underwater robots. The Eighteenth International Offshore and Polar Engineering Conference pp 382–388

Hanai A, Choi S, Marani G, Rosa K (2009) Experimental validation of model-based thruster fault detection for underwater vehicles. In: Robotics and Automation, 2009. ICRA '09. IEEE International Conference on, pp 194–199, DOI 10.1109/ROBOT.2009.5152425

Hanai A, Marani G, Choi SK (2010) Automatic fault-accommodating thrust redistribution for a redundant auv. 5th JSME/RMD International Conference on Advanced Mechatronics

Healey A, Rock S, Cody S, Miles D, Brown J (1995) Toward an improved understanding of thruster dynamics for underwater vehicles. IEEE Journal of Oceanic Engineering pp 354–361, DOI 10.1109/48.468252

Insaurralde C, Lane D (2012) Autonomy-assessment criteria for underwater vehicles. In: Autonomous Underwater Vehicles (AUV), 2012 IEEE/OES, pp 1–8, DOI 10.1109/AUV.2012.6380746

Isermann R (2005a) Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance. Springer Berlin Heidelberg

Isermann R (2005b) Model-based fault-detection and diagnosis–status and applications. Annual Reviews in control 29(1):71–85

Johansen T, Fossen T, Berge S (2004) Constrained nonlinear control allocation with singularity avoidance using sequential quadratic programming. Control Systems Technology, IEEE Transactions on 12(1):211–216, DOI 10.1109/TCST.2003.821952

Karras G, Bechlioulis C, Abdella H, Larkworthy T, Kyriakopoulos K, Lane D (2013) A robust sonar servo control scheme for wall-following using an autonomous underwater vehicle. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on, pp 3893–3898, DOI 10.1109/IROS.2013.6696913

Karras G, Bechlioulis C, Nagappa S, Palomeras N, Kyriakopoulos K, Carreras M (2014) Motion control for autonomous underwater vehicles: A robust model-free approach. In: Robotics and Automation (ICRA), 2014 IEEE International Conference on, pp 6529–6534, DOI 10.1109/ICRA.2014.6907822

Kim J, Han J, Chung WK, Yuh J, Lee PM (2005) Accurate and practical thruster modelling for underwater vehicles. In: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on, pp 175–180, DOI 10.1109/ROBOT.2005.1570115

Lane DM, Maurelli F, Larkworthy T, Caldwell D, Salvi J, Fox M, Kyriakopoulos K (2012) PANDORA: Persistent autonomy through learning, adaptation, observation and replanning. Proceedings of 3rd IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles pp 367–372, DOI 10.3182/20120410-3-PT-4028.00061

Maurelli F, Larkworthy T, Lane D, Karras G, Bechlioulis C, Kyriakopoulos K (2013) Pose-based and velocity-based approaches to autonomous inspection of subsea structures. In: Proceedings of IEEE-MTS Oceans'13, San Diego, USA

Mendonça LF, Sousa J, da Costa JS (2009) An architecture for fault detection and isolation based on fuzzy methods. Expert systems with applications 36(2):1092–1104

Miguelaez E, Patron P, Brown K, Petillot Y, Lane D (2011) Semantic knowledge-based framework to improve the situation awareness of autonomous underwater vehicles. Knowledge and Data Engineering, IEEE Transactions on 23(5):759–773, DOI 10.1109/TKDE.2010.46

Mišković N, Barišić M (2005) Fault detection and localization on underwater vehicle propulsion systems using principal component analysis. IEEE ISIE 2005 Conference pp 1721–1728

Omerdic E, Roberts G (2004) Thruster fault diagnosis and accommodation for open-frame underwater vehicles. Control Engineering Practice 12(12):1575 – 1598, DOI http://dx.doi.org/10.1016/j.conengprac.2003.12.014

Sarkar N, Podder T, Antonelli G (2002) Fault-accommodating thruster force allocation of an auv considering thruster redundancy and saturation. IEEE Transactions on Robotics and Automation pp 223–233, DOI 10.1109/TRA.2002.999650

Seto M (2012) Marine Robot Autonomy. SpringerLink : Bücher, Springer

Strang G (2006) Linear Algebra and Its Applications. Thomson, Brooks/Cole

Tang L, Zhang B, DeCastro J, Hettler E (2011) An integrated health and contingency management case study on an autonomous ground robot. In: Control and Automation (ICCA), 2011 9th IEEE International Conference on, pp 584–589, DOI 10.1109/ICCA.2011.6137995

Valeyrie N, Maurelli F, Patron P, Cartwright J, Davis B, Petillot Y (2010) Nessie V Turbo: a new hover and power slide capable torpedo shaped AUV for survey, inspection and intervention. In: AUVSI North America 2010 Conference

Vijayakumar S, D'Souza A, Schaal S (2005) LWPR: A scalable method for incremental online learning in high dimensions

Wang J (2012) Fault diagnosis of underwater vehicle with fnn. In: Intelligent Control and Automation (WCICA), 2012 10th World Congress on, pp 2931–2934, DOI 10.1109/WCICA.2012.6358371

Wang L, Tan K, Chew C (2006) Evolutionary Robotics: From Algorithms to Implementations. World Scientific series in robotics and intelligent systems, World Scientific Publishing

Willcox J, Bellingham J, Zhang Y, Baggeroer A (2001) Performance metrics for oceanographic surveys with autonomous underwater vehicles. Oceanic Engineering, IEEE Journal of 26(4):711–725, DOI 10.1109/48.972114

Yang K, Yuh J, Choi S (1998) Experimental study of fault-tolerant system design for underwater robots. In: Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on, vol 2, pp 1051–1056 vol.2, DOI 10.1109/ROBOT.1998.677229